Universidad Carlos III

ESCUELA POLITÉCNICA SUPERIOR

Ingeniería Técnica de Telecomunicación: Telemática

Aplicación de Técnicas de Optimización Convexa al Modelado de Entornos de Emergencia

---

Autor: Pablo de la Fuente Nuez

Tutor: Isaac Seoane Pujol

Resumen

Las redes ad hoc no requieren ninguna infraestructura previa para desplegarse, de ahí viene su potencial. Además, están cobrando un gran protagonismo gracias a la proliferación de dispositivos de uso cotidiano capaces de propiciarlas. Estas redes están formadas por un número de dispositivos inalámbricos de gran autonomía energética, capaces de crear redes de este tipo para comunicarse entre sí y/o con otras redes de datos.

Poder analizar, estimar y prever el comportamiento de este tipo de redes en diferentes escenarios (rurales, urbanos, de emergencias, etc.) con un elevado grado de realismo, es muy importante para estudiar distintos mecanismos de optimización (tráfico o capacidad de enlaces) que reduzcan la congestión de la red o hagan un uso eficiente de los recursos. Tener una herramienta que permita simular y trabajar con todos los ámbitos implicados en estos escenarios (movilidad, encaminamiento y optimización) resultaría de gran utilidad.

En este proyecto se sientan las bases para tratar la necesidad descrita, estudiando diferentes modelos de movilidad humana, algoritmos de encaminamiento y su optimización utilizando una técnica de optimización convexa, se desarrolla el escenario de simulación y se describen sus resultados.

Palabras Clave Modelos de Movilidad, Optimización Convexa, Redes Ad-Hoc

Abstract

The potential of Ad-hoc networks come from its lack of any kind of infrastructure to be deployed. Nowadays, this networks are becoming increasingly important thank to the growth of wireless devices with great energy autonomy which are able to create such networks to communicate between them or with other data networks.

It is important to analyze, estimate and foresee the behavior of this kind of networks in different scenarios (e.g. rural, urban or emergency environments) with the maximum degree of realism, to apply diverse optimization mechanisms (e.g. traffic or link capacity) which either decrease congestion in the network or make an efficient resource consumption. Having a tool that allows to simulate and to work in all this areas for this environments (human mobility, routing and optimization) would be very useful.

In this project we try to solve this requirements, by means of the study of several human mobility models, routing algorithms and their optimization by using convex optimization techniques. We also deploy the testbed of the simulation and finally we show the results obtained.

Keywords Mobility Models, Ad hoc Networks, Convex Optimization

Agradecimientos

Primero me gustaría dar las gracias a mi familia. En especial, a mis padres, por todo el apoyo que han depositado en mí, gracias a ellos soy lo que soy. A mi hermano Javi por su apoyo incondicional y ser una gran persona, y a mi prima Elena, por interesarse siempre por mí, un beso muy fuerte para ella.

También quiero agradecer a mi tutor Isaac por aceptarme como proyectando suyo, por sus ganas, su esfuerzo y su apoyo. Este trabajo sale gracias a él.

A la gente de atletismo, a mis entrenadores, Rosendo y Jose Luis Calvo, gracias por aquellos años tan buenos y enseñarme que con esfuerzo y constancia se puede conseguir todo lo que uno se propone. Y a todos los amigos que hice durante ese tiempo, HSA Team no se olvida.

A mis amigos del "cole", Alberto, Alvaro, Anita, Ire, Iris, Javi, Lu y Manu por toda una vida juntos, y las que nos quedan. A Carlitos, Pichu, Pipo y Puli, porque aunque no nos veamos tanto, siempre me acuerdo de vosotros. A todos, gracias.

Por último me gustaría dar las gracias a mis compañeros, convertidos en amigos, de la universidad. A la peque, Aida, por el cariño y apoyo que me has dado, gracias. A rober, por aguantarme tanto tiempo y por dejarme aguantarte a ti también. A Jesús, un tío al que aprecio, tomarse unas cañas contigo es otra cosa. A Carlitos, por conseguir hacerme reír en todo momento, eres un grande. A Adal, por hacer de cualquier situación una historia a recordar. A Dieguete, porque aunque ya no te dejes ver tanto como antes, en el primer recuerdo que tengo de este sitio sales tú. A Samu y a Ele, porque junto con dieguete, sois de los pesados que siempre habéis estado ahí. A Lisar y Virgi, por los buenos momentos y el apoyo que me habéis dado. Y especialmente, a Miguel y a Chemi, dos de las mejores personas que conozco, compañeros de beca, y sobre todo, buenos amigos y a Deivid, por toda la ayuda incondicional que me ha dado siempre.

A todos, muchísimas gracias por hacer que este espacio se quede corto.

Índice general

Índice de Figuras

Índice de Tablas

Capítulo 1

Introducción

1.1. Motivación

En general, las redes ad hoc son redes formadas por una cantidad de nodos con ciertas limitaciones que se comunican entre ellos sin necesidad de ninguna infraestructura, transmitiendo la información de unos a otros. Esto hace posible que el despliegue sea más rápido y económico que el de las redes fijas.

Las redes ad hoc han cobrado una gran importancia debido a su movilidad y fácil distribución. El abaratamiento de costes de fabricación y al desarrollo tecnológico que ha permitido reducir su consumo energético y tamaño ha hecho que el crecimiento de estas redes sea muy rápido. A día de hoy existen muchos estudios que permiten que estas redes se encuentren en constante evolución de sus prestaciones. Estos estudios se centran en campos como la calidad de servicio ofrecida a los usuarios, la seguridad, la autonomía de las baterías de los terminales, etc.

Se trata por tanto de redes con una topología cambiante, en las que cada nodo puede jugar el papel de emisor, receptor o intermediario, es decir, no existen a priori roles específicos para cada nodo. Dada la movilidad que plantean este tipo de redes, se tiene que tener especial cuidado con las capacidades de los enlaces y mecanismos de encaminamiento para obtener el mejor funcionamiento posible de este tipo de redes.

Este tipo de redes pueden ser utilizadas en distintos entornos, situaciones y escenarios. A continuación se detallan algunos ejemplos:

- Militares: el desarrollo militar en multitud de ocasiones es el que ha promovido mu-

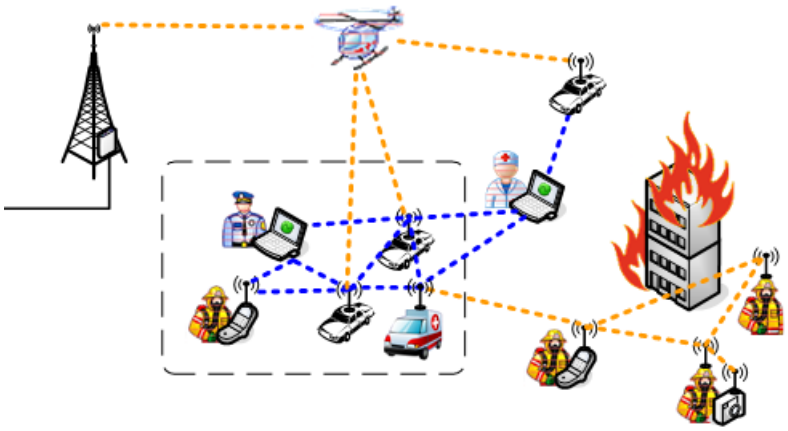


Figura 1.1: Ejemplo de red ad hoc en situaciones de emergencia [1].

chos avances tecnológicos y las redes ad hoc no son una excepción. En estos entornos es necesario establecer una comunicación entre distintas unidades, vehículos o centros de mando sin necesidad de utilizar una infraestructura fija.

- Emergencias: cuando los equipos de emergencia, rescate o salvamento tienen que actuar rápidamente, no existe la posibilidad de instalar una infraestructura fija. Desplegar una red ad hoc es una solución rápida y eficaz para muchas de estas situaciones.

- Civiles: la formación de redes ad hoc podría también servir como acceso inalámbrico público en zonas urbanas, proporcionando un rápido despliegue y una extensión de la cobertura. Algunos de los puntos de acceso proporcionarían también pasarelas a través de las cuales los usuarios se podrían conectar a una red fija.

- Industriales o agrícolas: este tipo de redes son de gran utilidad en estos ámbitos, haciendo uso de estas redes como redes de sensores, para monitorizar temperatura, luz, humedad, etcétera.

- A nivel local: las redes ad hoc que enlazan dispositivos móviles pueden ser usadas para difundir y compartir información entre los participantes de una conferencia. También pueden servir para redes domésticas, donde los dispositivos se comunican directamente para intercambiar información, como audio/vídeo, alarmas o actualizaciones de configuración.

1.2. Objetivos del Proyecto

En este proyecto se sientan las bases para el desarrollo de un simulador completo de redes ad hoc, que abarca desde el movimiento de los nodos por el escenario hasta una posible optimización de la red en cuanto a demandas, tráficos y capacidades entre nodos. Con ello se conseguirá una herramienta de estudio que permita trabajar independientemente en las diferentes partes de una simulación de este tipo.

En base al objetivo principal, existen un conjunto de hitos a alcanzar separados en tres partes diferenciadas:

- Estudio de los modelos de movilidad: comprender los modelos existentes, el funcionamiento de los mismos y los criterios a tener en cuenta para su diseño e implementación.
- Estudio del encaminamiento en redes ad hoc: análisis de los algoritmos, implementación e integración en el simulador.
- Optimización de los servicios ofrecidos de acuerdo a funciones de coste.

Un objetivo añadido de este proyecto parte del interés de profundizar en una herramienta como es el entorno de desarrollo MATLAB y su lenguaje de programación para este tipo de escenarios y situaciones.

1.3. Contenido

Este documento se aleja de la estructura habitual de este tipo de trabajos, intentando facilitar la lectura y entendimiento de la memoria. Se ha dividido el documento en tres grandes bloques (correspondientes a los capítulos dos, tres y cuatro) que contendrán su propia arquitectura orientada a los contenidos que ofrece y las conclusiones que se extrae de cada uno. La memoria también incluye un capítulo de introducción, uno de integración y conclusiones.

En el primer bloque (segundo capítulo) se lleva a cabo todo el desarrollo de la parte del simulador relacionada con la movilidad, comprendido entre el estudio de los distintos modelos existentes, y la implementación de modelos de movilidad con características

particulares, pasando por el análisis de diseño de los modelos, parámetros de los nodos, etc. En el tercer capítulo, se estudia, diseña e implementa toda la parte concerniente al encaminamiento. Se trata por tanto del segundo gran bloque del simulador. Finalmente el cuarto capítulo engloba la parte final de la simulación y corresponde con el tercer bloque diferenciado del simulador. Consiste en un estudio de las características de la optimización convexa, y un diseño e implementación para la integración dentro de el proyecto.

A continuación, en el quinto capítulo se explica la integración de los tres bloques expuestos en los anteriores capítulos, ofreciendo una visión general del simulador, y planteando mediante un ejemplo real, una simulación completa y sus resultados.

Para terminar, en el sexto capítulo se hace una recapitulación de las conclusiones obtenidas tras la realización del proyecto y se describen las posibles líneas de trabajo futuro para la mejora y evolución de la plataforma propuesta.

Se añaden por último, un apéndice en el que se realiza un análisis de los costes económicos y temporales relacionados con las distintas fases de desarrollo del proyecto. Otro apéndice que detalla las estructuras de datos que maneja el simulador y un apéndice final para detallar las principales partes del código implementadas.

Capítulo 2

Modelado de Movilidad Humana

Este capítulo trata del análisis y diseño de modelos de movilidad humana. En él se describen algunos de estos modelos y las implementaciones desarrolladas de forma práctica.

2.1. Introducción a los Modelos de Movilidad

Es de gran importancia, a la hora de realizar simulaciones de redes ad hoc, definir y estudiar patrones de movimiento de los nodos, ya sean sensores o dispositivos móviles en personas o vehículos.

La implantación de este tipo de redes se está extendiendo cada vez más, gracias en parte a que una mayoría de la población lleva consigo dispositivos capaces de propiciarlas.

Surge, por tanto, la conveniencia de tener buenos modelos para estudiar patrones de movimiento permitiendo realizar cálculos y estimaciones adecuadas, como la capacidad de los enlaces, necesidades de distancia, autonomía, etc.

En este capítulo se realiza un estudio teórico de los diferentes modelos de movilidad existentes y sus características más importantes. En todos ellos, se busca un compromiso entre simplicidad y realismo, ya que el coste computacional de simular el movimiento de manera individual de una gran cantidad de nodos puede sobrepasar la capacidad de cómputo de la que se dispone. Sin embargo no conviene dejar de lado la necesidad de diseñar modelos lo suficientemente reales como para poder obtener datos útiles.

A nivel práctico, se presentan los modelos incorporados al simulador y su funciona-

miento, ofreciendo una panorámica del comportamiento de los nodos de manera individual.

En conclusión, se pretende dar una visión de los modelos existentes, cómo utilizarlos en simulaciones y su integración dentro este proyecto.

2.2. Estudio del Estado del Arte

2.2.1. Fundamentos del Modelado

Existen distintos modelos de movilidad, de los cuales se detallan en esta sección, aquellos más populares y alguna propuesta menos conocida pero interesante.

Para facilitar su compresión, a continuación se detallan los principales parámetros necesarios para crear un modelo de movilidad. Éstos no son los únicos, ni han de estar presentes en todos los modelos.

- Distancia: (recorrida por un nodo) puede expresarse de dos formas: indicando la propia distancia (en media o individualmente) o indicando la posición de destino a la que debe llegar el nodo.
- Tiempo de pausa: utilizado en simulaciones en las que el nodo, una vez alcanzado el destino, final o parcial, o expirado un temporizador, se detenga.
- Velocidad: (de un nodo en movimiento) está determinada entre un máximo y un mínimo.
- Dirección de salida: forma en la que un nodo elige la dirección para llegar al destino. Algunos modelos no hacen uso de este parámetro, obteniendo el destino mediante coordenadas.
- Aceleración/deceleración: utilizado en los modelos más complejos a la hora de implementar el movimiento de los nodos.

En la figura 2.1 se da una visión global de las características principales de los modelos de movilidad. El objetivo a la hora de modelar está enfocado en realizar simulaciones

o descripciones analíticas. Por lo general en las simulaciones se utiliza el dominio de tiempo discreto, que permite un mayor control a la hora de analizar los datos, por otro lado, el dominio de tiempo continuo es utilizado para las descripciones, ya que modelan mejor escenarios reales. Un modelo de movilidad puede presentarse en una, dos o tres dimensiones del espacio.

El grado de aleatoriedad es importante a la hora de clasificar los modelos de movilidad. En función de este existen los modelos:

- Deterministas: aquellos de los que se tiene un conocimiento suficiente para saber de antemano cómo va a comportarse el sistema.
- Aleatorios: en los que se desconoce el comportamiento o criterios de decisión que toman los nodos para moverse.
- Híbridos: que son una mezcla de los anteriores.

La mayoría de los modelos de simulación son híbridos, ya que definen barreras estructurales, tipo paredes u obstáculos del escenario.

Existen tres tipos de movimiento para los modelos de movilidad que son individual, grupal o con un líder:

- En los modelos individuales (no correlados) los nodos no condicionan su movimiento a lo que ocurra a su alrededor, es decir, no están afectados por los nodos vecinos.
- El movimiento grupal se refiere a aquellas situaciones en la que los nodos toman decisiones condicionadas, como por ejemplo: zonas del escenario con características especiales de movimiento, asignación de varios nodos con parámetros distintos al resto o control de proximidad de nodos.
- Los modelos de movilidad con líder consisten en asignar a uno de ellos el papel de jefe (pueden ser uno o varios) y marcar el camino que seguirán los demás.

A continuación se presentan los modelos de movilidad humana estudiados en este proyecto.

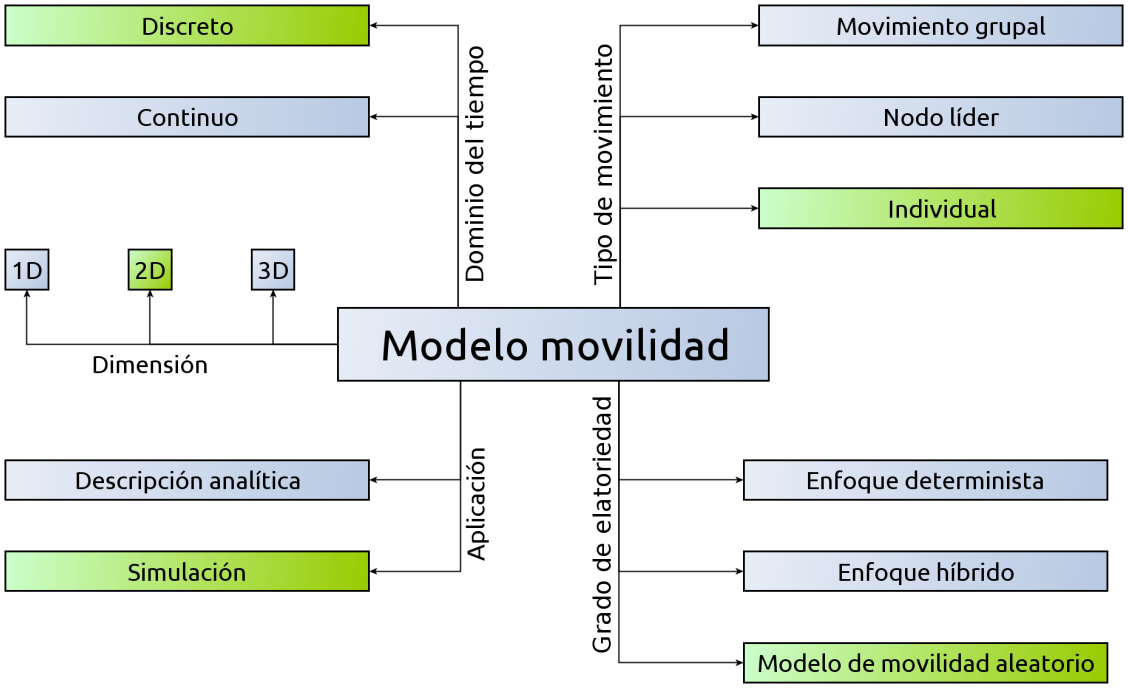


Figura 2.1: Esquema general de los modelos de movilidad.

2.2.2. Modelo Random Walk

Se trata probablemente del modelo más sencillo de los que se van a describir [2]. Se utiliza para simular movimientos erráticos de la naturaleza. También es utilizado para movimientos de personas, cuando el estudio no requiera un modelo con unas prestaciones muy altas y se busque un modelado sencillo y rápido.

Su funcionamiento se basa en un nodo móvil que se desplaza a una posición aleatoria dentro del escenario, a partir de una dirección y una velocidad elegidas aleatoriamente. Estos parámetros cambian a intervalos fijos de tiempo o por algún otro criterio, como por ejemplo, la distancia recorrida. Los valores de velocidad y dirección se seleccionan de forma aleatoria a partir de distribuciones uniformes y han de estar contenidos entre $[v_{min}, v_{max}]$ y $[0, 2\pi]$ respectivamente.

De manera que el proceso que sigue un nodo bajo este modelo sigue los siguientes pasos:

- Se escoge una dirección, dada a partir de una distribución uniforme, $U[0, 2\pi]$.

- Se escoge una velocidad a partir de $U[v_{min}, v_{max}]$.
- El nodo se dirige en esa dirección y con esa velocidad, hasta que ocurra la condición de cambio. Normalmente tras expirar un temporizador (también se puede usar condiciones de distancia).
- Una vez ocurre dicha condición, el nodo es reasignado con una nueva velocidad y dirección.

2.2.3. Modelo Random Waypoint

Este modelo [3] añade respecto al anterior la característica de dotar a los nodos con un tiempo de pausa. El nodo escoge la siguiente posición dentro del escenario y la velocidad a partir de una distribución uniforme $U[v_{min}, v_{max}]$. Al expirar el tiempo de pausa se dirige al destino y una vez alcanzado este, el nodo vuelve a esperar otro tiempo de pausa antes de volver a moverse. Este modelo de movilidad es el más popular para la mayoría de las simulaciones gracias a su sencillez y validez.

El proceso que sigue un nodo bajo este modelo sigue los siguientes pasos:

- Se escoge una posición, dada a partir de una distribución uniforme (elije las nuevas coordenadas de destino).
- Se escoge una velocidad a partir de $U[v_{min}, v_{max}]$.
- El nodo se dirige a esa posición y con esa velocidad en línea recta hasta alcanzar el destino (parcial o final).
- Una vez se alcanza, el nodo iniciará su estado de pausa (fijo o aleatorio).
- Finalizado el estado de pausa, el nodo es reasignado con una nueva posición y velocidad.

2.2.4. Modelo Random Direction

Modelo sin mecanismo de pausado. La velocidad de los nodos se toma a partir de una distribución uniforme (como en los anteriores modelos). Presenta una nueva forma

de gestionar el cambio de dirección: se cambia de dirección a intervalos fijos de tiempo o a partir de distribuciones exponenciales [4], en distintos instantes que el evento de cambio de velocidad [5].

Otra implementación común es forzar al nodo a avanzar hasta el borde de la pared una vez escogidas su velocidad y dirección [6], surgiendo la necesidad de realizar un control de los rebotes. El fenómeno de rebote se da cuando un nodo llega a un borde del escenario. En este punto, si no se utilizan condiciones especiales para la elección de la nueva dirección se puede dar lugar a la acumulación de los nodos en los bordes y/o salida del nodo del escenario, saliéndose de la zona estudiada.

Se trata de un modelo con distintas implementaciones bajo el mismo nombre.

El funcionamiento general que sigue un nodo bajo este modelo sigue los siguientes pasos:

- Se escoge una dirección, dada a partir de una distribución uniforme, $U[0, 2\pi]$.
- Se escoge una velocidad a partir de $U[v_{min}, v_{max}]$.
- El nodo se dirige en esa dirección y con esa velocidad, hasta que ocurra la condición de cambio.
    - Si la condición se refiere a la dirección, el nodo tomará una nueva y continuará con la velocidad asignada y la nueva dirección.
    - Si la condición se refiere a la velocidad, el nodo tomará una nueva y continuará en la misma dirección y distinta velocidad.
- El proceso se repite de esta manera hasta que finaliza la simulación.

2.2.5. Modelo Smooth Random Mobility

Modelo de movilidad con mayor grado de complejidad que ofrece un nivel de realismo superior [4]. Se trata de una particularización del modelo general *Random Direction* en dos dimensiones que aporta algún detalle novedoso.

La elección del tiempo de cambio de velocidad y de dirección hace uso de dos procesos estocásticos diferentes. El modelo tiene en cuenta el valor previo que llevaban los

nodos a la hora de tomar una nueva velocidad o dirección. De esta forma se consigue suavizar los giros que describen los nodos, evitando cambios bruscos de dirección, y los cambios de velocidad aleatorios, que pueden originar un comportamiento irreal.

Dado que el enfoque final de este proyecto está basado en eventos de tiempo, a continuación se detalla la definición discreta de este modelo utilizando la siguiente notación (también existe para dominios del tiempo continuo):

- n : hace referencia al intervalo de tiempo en el que nos encontramos (en vez de instantes de tiempo t).
- Δn : será el tiempo fijo del intervalo temporal.
- $n/\Delta n$: representa el número del intervalo en el que se encuentra la simulación.

Control de la velocidad:

Es importante detallar la gestión de la velocidad que propone este modelo, puesto que hace uso de un nuevo concepto no visto hasta ahora: **velocidades preferentes** (v_{pref}), gracias a las cuales los nodos toman con mayor probabilidad alguna de las velocidades seleccionadas por el diseñador. Un cambio de velocidad se realiza mediante una aceleración o deceleración hasta alcanzar la velocidad objetivo.

Para la elección de la nueva velocidad se utiliza una función de distribución como esta:

$$p(v) = \begin{cases} p(v = v_{pref1})\delta(v - v_{pref1}) & \text{si } v = v_{pref1} \\ p(v = v_{pref2})\delta(v - v_{pref2}) & \text{si } v = v_{pref2} \\ \dots & \\ p(v = v_{prefV})\delta(v - v_{prefV}) & \text{si } v = v_{prefV} \\ \frac{1-p(v_{pref})}{v_{max}} & \text{si } 0 \leq v \leq v_{max} \end{cases}$$

$$p(v_{pref}) = \sum_{i=1}^{V} p(v_{pref_i}) < 1$$

La ecuación dará la probabilidad de selección de las velocidades por cada nodo. No existe un límite en el número de las velocidades preferentes, siempre y cuando se cumpla la siguiente condición:

$$p(v_{pref}) = \sum_{i=1}^{V} p(v_{pref_i}) < 1$$

Condición básica por la cual, la suma de la probabilidades de las velocidades preferentes no puede ser mayor que uno, es decir, cada una de esas velocidades preferentes viene asociada a una probabilidad de que el nodo la seleccione, y el conjunto de todas ellas no puede rebasar el cien por cien. Las velocidades restantes, acotadas por el límite superior a v_{max}, podrán ser seleccionadas con una probabilidad uniforme a partir de la probabilidad restante.

La aceleración con la que un nodo llega a la velocidad objetivo viene dada por una distribución uniforme. El nodo irá acelerando o decelerando hasta alcanzar la velocidad objetivo, momento en el que pasa a velocidad constante.

Por lo tanto en cada intervalo de tiempo, la velocidad se recalcula siguiendo la siguiente expresión:

$$v[n] = v[n - \Delta n] + a[n]\Delta n$$

De esta forma, un nodo que vaya a cambiar de velocidad lleva a cabo el siguiente proceso:

- Se escoge una nueva velocidad a partir de la probabilidad de selección $p(v)$.
- Se realiza un cálculo para analizar si hace falta acelerar o decelerar y se toma una aceleración a partir de las distribuciones uniformes $U[a_{min}, 0)$ y $U(0, a_{max}]$ en base a lo que corresponda.
- El nodo cambia de velocidad en cada intervalo hasta alcanzar la velocidad objetivo.
- Un vez alcanzada el nodo deja de acelerar y se mantiene a esa velocidad hasta alcanzar el destino (parcial o final).

Control de la dirección:

A continuación se describe el mecanismo de control de dirección de este modelo. Inicialmente los nodos toman una dirección a partir de una distribución uniforme $U[0, 2\pi]$.

En cada intervalo de tiempo que corresponda hacer un cambio de dirección (decidido mediante un proceso estocástico) se inicia el siguiente proceso:

- Se escoge una nueva dirección a partir de la distribución uniforme de $U[0, 2\pi]$.
- Se realiza un cálculo para hacer el giro del nodo en el sentido que menos variación provoque.
- Luego se seleccionan uno o varios *tiempos de curva* a partir de una distribución uniforme que hará que el nodo vaya girando de forma suave hacia la dirección objetivo.
- El proceso se detiene cuando el nodo alcance dicha dirección o cuando un nuevo evento de cambio ocurra.

El radio de la curva descrito por el nodo también está controlado, debido a que depende de forma lineal con la velocidad del nodo en ese instante. A mayor velocidad, mayor será la curva realizada por el nodo.

2.2.6. Modelo Truncated Levy Walk

El último modelo de movilidad documentado en este proyecto se basa en una implementación del grupo de investigación *Mobility-aware Networking*[1] de la Universidad de Carolina del Sur[2] [7]. El interés de este modelo está en la validación realizada por ellos mismos a partir de trazas reales de movimientos de personas en distintos escenarios. En base a los resultados que obtuvieron han planteado un modelo con las siguiente formulación:

- Distancias de trayecto: siguen una distribución de ley de potencias truncada:

$$p(l) \sim |l|^{-(1+\alpha)},\ l < l_{max}$$

- Tiempos de pausa: siguen otra distribución de ley de potencias truncada:

$$\psi(t) \sim t^{-(1+\beta)},\ 0 < t < t_{max}$$

[1] http://research.csc.ncsu.edu/netsrv/?q=content/mobility-aware-networking-group
[2] http://www.ncsu.edu/

- Otras consideraciones:
  - La elección de ángulos de giro sigue una distribución uniforme.
  - La velocidad incrementa a medida que lo hace la longitud.

Ley de potencias:

El modelo sigue una distribución de ley de potencias que provoca un disminución de la frecuencia (probabilidad de tomar un valor) según un exponente (α y β en el modelo), cuando la variable aleatoria (l o t) aumenta. La trayectoria descrita por un nodo a gran escala resulta ser una combinación de agrupaciones de trayectos con largos tramos entre ellos (llamados *vuelos*).

El truncado viene dado por $l < l_{max}$. Debido a que sigue una distribución basada en una función de cola pesada, es decir, se distribuye mediante una ley potencial, el dominio de esta es $[0, \infty]$

Por lo tanto, el funcionamiento que sigue un nodo bajo este modelo sigue los siguientes pasos:

- Se escoge una dirección, dada a partir de una distribución uniforme, $U[0, 2\pi]$.
- Se escoge una velocidad a partir de $U[0, v_{max}]$.
- Se escoge una distancia a partir de $p(l) \sim |l|^{-(1+\alpha)},\ l < l_{max}$.
- El nodo se dirige en esa dirección y con esa velocidad, hasta que ocurra la condición de cambio, el nodo llega al destino.
- En ese momento el nodo selecciona un tiempo de pausa a partir de $\psi(t) \sim t^{-(1+\beta)},\ 0 < t < t_{max}$.
- Un vez expirado el tiempo de pausa el nodo es reasignado con unos nuevos parámetros y se vuelve a poner en movimiento.

El nodo al seguir distribución basada en una cola pesada, irá tomando (después de una distancia inicial) valores sucesivamente más pequeños, hasta que se alcance el truncado, que hace que la siguiente distancia del nodo se mayor que la anterior (da un salto

Modelo	Pausa	Rebotes	Memoria	Correlación
Random Walk	no	no	no	no
Random Waypoint	✓	•	no	no
Random Direction	no	✓	no	no
Smooth Random	no	•	✓	no
Truncated Levy Walk	✓	•	✓	no
Pursue Model	✓	•	•	✓
RPGMV	•	•	✓	✓

Tabla 2.1: Tabla comparativa de las principales características de los modelos de movilidad

en magnitud). Esto hace que el comportamiento del nodo sea el descrito anteriormente (trayectos largos y agrupaciones de trayectos cortos).

2.2.7. Modelos no individuales:

Los modelos descritos anteriormente son exclusivamente individuales, es decir, no están correlados. Pero existen otros modelos que merecen ser destacados [8].

- *Pursue model*: Modelo de tipo líder. Los nodos persiguen a un único nodo objetivo, que puede estar parado o no. El movimiento de los nodos restantes esta condicionando a la persecución del líder de forma pseudoaleatoria. Se trata de un comportamiento muy común en rastreo de personas.

- *Reference point group mobility vector (RPGMV)*: Basado en un movimiento grupal. Cada grupo tiene un centro lógico, que define todo el comportamiento de los nodos pertenecientes a él (localización, velocidad, dirección, aceleración, etc.). La trayectoria del grupo se determina mediante un camino configurado para el centro.

Una comparativa de las características de estos y los anteriores puede verse en la tabla 2.1

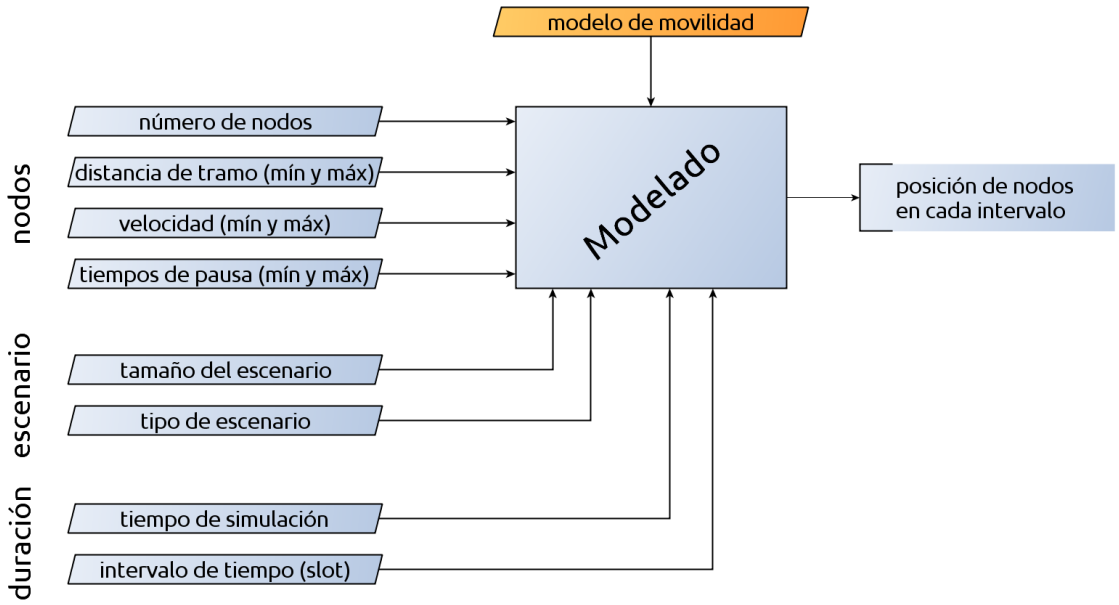


Figura 2.2: Esquema general del bloque de modelado

2.3. Descripción del Bloque de Modelado de Movimientos

A continuación se describen las decisiones de diseño tomadas para la implementación de los modelos de movilidad integrados en el simulador.

2.3.1. Estructura del Bloque

Este primer bloque engloba la parte del simulador encargada de implementar modelos de movilidad para simular escenarios reales. Ofrece una herramienta para permitir diseñar una serie de escenarios dados unos parámetros iniciales.

Los parámetros utilizados para realizar el modelado de movimiento se presentan en la figura 2.2 y como se puede ver en la misma, se clasifican en tres tipos: los parámetros referentes a los nodos de manera individual, los utilizados para la configuración del escenario y finalmente el tiempo de simulación y tamaño del intervalo de tiempo. Todos ellos son utilizados para al menos uno de los modelos implementados en este proyecto.

Parámetro	Dimensión	Uso
N	`N`	Número de nodos
P	`[N×1]`	Tiempo restante pausado
D	`[N×1]`	Distancia por recorrer
V	`[N×1]`	Velocidad de los nodos en movimiento
DIR	`[N×2]`	Dirección de los nodos en movimiento
POS	`[N×2]`	Posición en el escenario

Tabla 2.2: Tabla resumen con los parámetros del Random Wapoint modificado.

2.3.2. Modelos de movilidad implementados

Se han integrado en el simulador tres modelos de movilidad. Se tratan, en algunos casos, de variantes de los modelos definidos en el estado del arte. A continuación se detallan las características más importantes de cada uno de ellos.

A. Implementación del modelo Random Wapoint modificado:

El primer modelo diseñado es una variante del Random Waypoint [3]. La definición general propone al nodo una nueva posición a la que dirigirse en cada intervalo que corresponda cambiar dicho parámetro. Sin embargo, para esta modificación se ha rediseñado la propuesta para el mecanismo de selección de posición mediante una dirección, distancia y velocidad dadas. El nodo, al expirar su tiempo de pausa en el intervalo actual, obtendrá un nuevo valor para cada uno de estos parámetros aleatoriamente, a partir de distribuciones uniformes. No entra en pausa de nuevo hasta que recorra toda la distancia asignada.

El código posee una estructura para caracterizar a los nodos en cada instante de tiempo que consiste en un conjunto de matrices y vectores resumidos en la tabla 2.2. Siendo `N` el número de nodos de la simulación, se define un vector `P[N×1]` que indica el tiempo de pausa restante de cada nodo. Un vector `D[N×1]` para la distancia a recorrer, y uno `V[N×1]` para las velocidades. Además dos matrices, utilizadas para almacenar la dirección de los nodos en movimiento `DIR[N×2]`, y la posición actual de todos los nodos en un instante de tiempo, `POS[N×2]`.

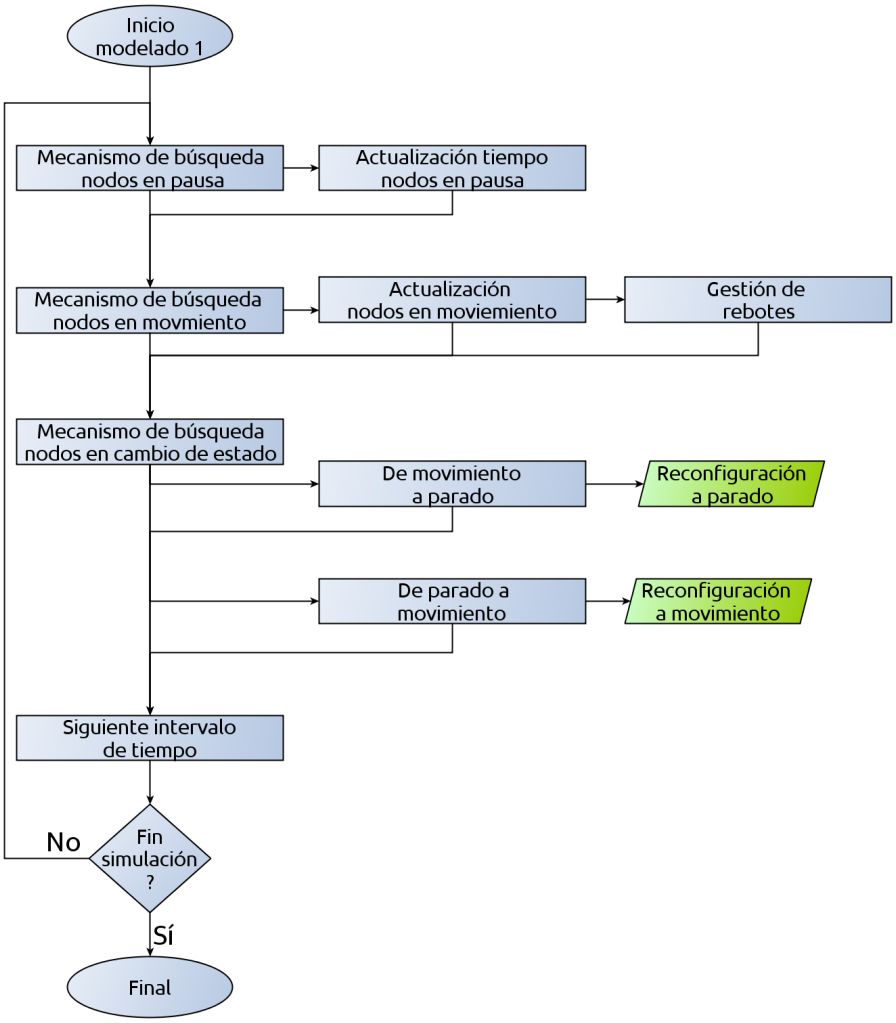


Figura 2.3: Funcionamiento del modelo Random Wapoint modificado.

El funcionamiento de este modelo se muestra en la figura 2.3 de forma detallada. Primero se actualizan los nodos que no cambian de estado, modificando el tiempo de pausa restante (en los nodos parados) o la distancia recorrida (en los nodos en movimiento). A continuación, se reasignan nuevos valores para las nodos que cambien de estado en ese intervalo de tiempo.

Respecto a la gestión de los nodos que llegan a un borde, se opta por establecerles como nodos en movimiento en el siguiente intervalo de tiempo, evitando problemas con múltiples rebotes (si se efectuase el rebote en ese mismo instante de tiempo) y posibles fenómenos de acumulación en los bordes (cambiando el estado del nodo a parado).

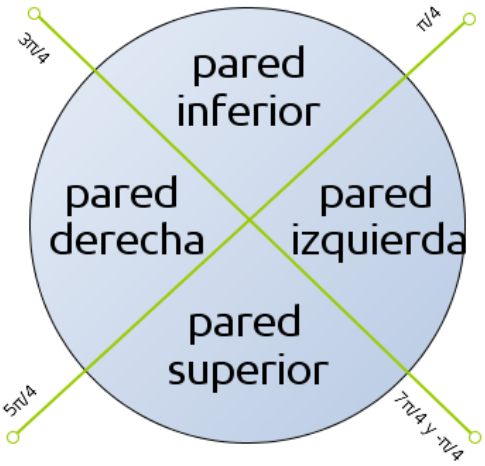


Figura 2.4: Rango de direcciones en los bordes.

La asignación de una nueva dirección a un nodo que esté situado en un borde, sólo tomará valores de una distribución uniforme acotada en función de la pared en la que se encuentre (ver figura 2.4). Impidiendo que la nueva dirección dirija al nodo contra la pared o paralelo al borde, es decir, trata de limitar el ángulo de salida dentro de un rango de $\pi/2$

La gestión del estado de pausa se ha diseñado de la siguiente manera: los nodos que salgan de este estado en el intervalo de tiempo actual, generan los nuevos datos de movimiento para comenzar en el siguiente, posibilitando así el mantenimiento de una estructura basada en eventos que permita un análisis más sencillo de la simulación completa.

B. Implementación del modelo Random Direction modificado:

El siguiente modelo está basado en el Random Direction [4] [6], se trata de una implementación que dota a los nodos de una mayor movilidad reduciendo los movimientos rectilíneos. Éstos no tienen una distancia o posición objetivo para entrar al estado de parado por lo que se ha tenido que diseñar un sistema de pausado de nodos.

Se hace uso de un parámetro utilizado exclusivamente al comienzo de la simulación para iniciar el número de nodos pausados. Indica el porcentaje de los nodos que se quieren tener parados. En este modelo se utiliza una distribución normal $G(\mu, \sigma)$, de media μ el

Parámetro	Dimensión	Uso
p	[1×1]	Porcentaje nodos en pausa
N	N	Número de nodos
P	[N×1]	Tiempo restante pausado
V	[N×1]	Velocidad de los nodos en movimiento
DIR	[N×2]	Dirección de los nodos en movimiento
POS	[N×2]	Posición en el escenario

Tabla 2.3: Tabla resumen con los parámetros del modelo de movilidad Random Direction modificado.

porcentaje deseado de nodos que se quieren tener pausados y desviación típica $\sigma = 0,1$, para controlar el número de nodos parados a lo largo de la simulación. En cada intervalo de tiempo se cuentan los nodos en movimiento y se estima una cantidad pseudoaleatoria de nodos a pausar para estar cerca de ese porcentaje.

Los nodos cambian de dirección y velocidad en pequeños intervalos de tiempo dentro del actual, evitando movimientos rectilíneos. Estos intervalos cortos se obtienen de una distribución exponencial truncada al tamaño del intervalo principal de la simulación y de media la mitad de este. Estos intervalos de tiempo son comunes para todos los nodos, no existe una distribución específica para cada uno de ellos.

El comportamiento de los nodos queda caracterizado en estado de pausa, por el vector P[N×1], y en movimiento por una dirección DIR[N×2] y una velocidad V[N×1] (tabla 2.3). El esquema de la figura 2.5 muestra la implementación del movimiento de los nodos en este modelo siendo el resto del funcionamiento similar al modelo Random Wapoint modificado.

Caracterización del comportamiento de los nodos: en estado de pausa, por el vector P[N×1], y en movimiento por una dirección DIR[N×2] y una velocidad V[N×1], tabla 2.3.

C. Implementación del modelo Truncated Levy Walk modificado:

Se trata de una implementación del Truncated Levy Walk [7]. Es un modelo diseñado e implementado por el grupo de investigación *Mobility-aware Networking*. Se ha llevado a

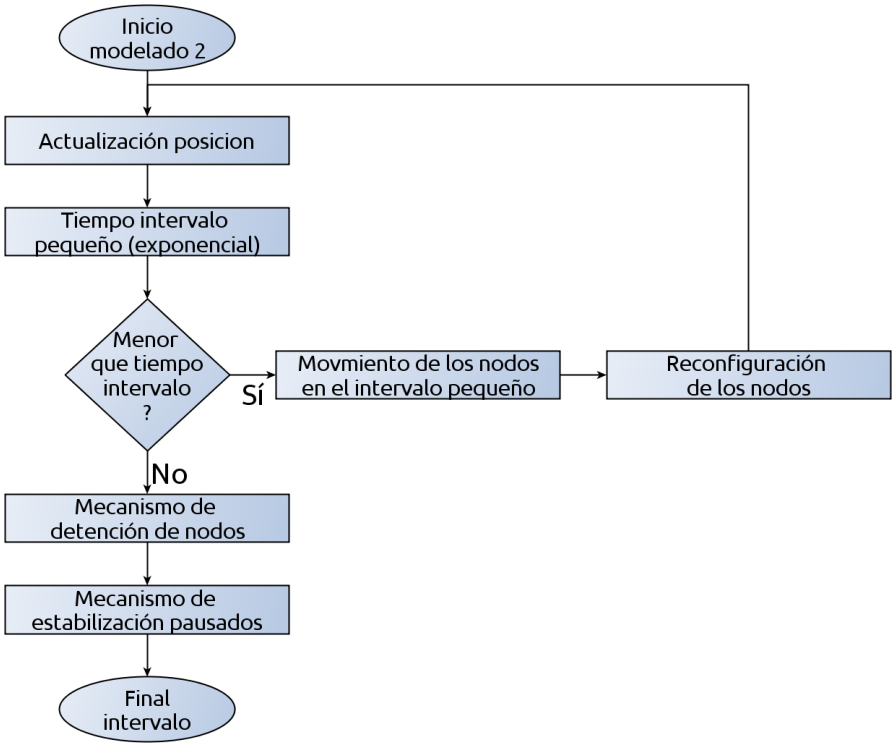


Figura 2.5: Funcionamiento de la parte de actualización de la posición en el modelo Random Direction modificado.

cabo una adaptación del modelo para integrarlo dentro de este proyecto ya que el funcionamiento interno difiere al de los otros.

Consiste en una función que tiene como parámetros de entrada los siguientes:

```
TLW_MATLAB(alpha,beta,size_max,s_min,s_max,duration,b_c)
```

- `alpha`: exponente característico de la distribución de la distancias de trayecto.
- `beta`: exponente característico de la distribución de los tiempos de pausa.
- `size_max`: tamaño del escenario (recinto cuadrado).
- `s_min` y `s_max`: tiempo mínimo y máximo de pausa.
- `duration`: tiempo total de simulación (minutos).
- `b_c`: condición de rebote (recinto cerrado o avenida comercial).

Por otro lado, como salida, devuelve una matriz de tres columnas. Las dos primeras columnas representan la posición del nodo y la tercera el instante de tiempo correspondiente a esa posición; y tantas filas como minutos tenga la simulación. Esto hace necesario que la función sea ejecutada una vez para cada nodo. La llamada a la función se ejecuta al inicio de la simulación, en la parte de inicialización.

Durante el trascurso de la simulación se seleccionan las posiciones de los nodos de esta matriz en los instantes de tiempo en que se está interesado (se cogen las filas de la matriz correspondientes). De esta forma el simulador sólo se encargará del encaminamiento y optimización (capítulos tres y cuatro respectivamente).

Por defecto la implementación del Truncated Levy Walk del grupo de investigación *Mobility-aware Networking*, no contempla el escenario de tipo avenida comercial, sólo permite trabajar con el recinto cerrado (rebote en todas las paredes, sin zonas de pausa). Fue necesaria una modificación del código de la función para desarrollar otros escenarios de prueba; a través del parámetro `b_c` se realiza la elección del tipo de escenario.

La implementación de la función hace necesario por tanto un sistema de almacenamiento debido a la ejecución múltiple (tantas ejecuciones como nodos haya en la simulación) que ésta necesita. Para ello se crea una estructura de datos que almacena el recorrido de todos los nodos a lo largo de la simulación, una matriz `POS_all` de tamaño $[t_{simu}, 2N]$, donde t_{simu} es los minutos de simulación y N es el número de nodos del sistema, y con estructura:

$$
\begin{array}{c}
\\
\\
instantes \\
\\
\\
\downarrow
\end{array}
\begin{pmatrix}
pos_{x_1} & pos_{y_1} & pos_{x_2} & pos_{y_2} & \cdots & pos_{x_N} & pos_{y_N} \\
pos_{x_1} & pos_{y_1} & pos_{x_2} & pos_{y_2} & \cdots & pos_{x_N} & pos_{y_N} \\
 & & & \vdots & & & \\
pos_{x_1} & pos_{y_1} & pos_{x_2} & pos_{y_2} & \cdots & pos_{x_N} & pos_{y_N}
\end{pmatrix}
$$

Las columnas tomadas a pares indican la posición de cada nodo, coordenada *x* y coordenada *y*, y las filas *instantes* marcan los instantes de tiempo en minutos durante toda la simulación. Cada una de las filas de la matriz representa la instantánea con la posición de todos los nodos en el escenario, y cada par de columnas ofrece el recorrido de un nodo durante toda la simulación.

Para comprender mejor esta estructura de almacenaje se presenta el siguiente ejem-

plo:

- *Simulación de tres minutos.*
- *Dos nodos en el sistema.*

Por tanto se ejecutará la función dos veces, devolviendo el recorrido de los dos nodos:

- *La primera ejecución devuelve el recorrido del primer nodo, por ejemplo:*
  (2,3) → (3,5) → (3, 7).
- *La segunda ejecución devuelve el recorrido del segundo nodo, por ejemplo:*
  (6,4) → (5,5) → (3, 6).

El simulador almacena en POS_all la siguiente información:

$$POS\_all = \begin{pmatrix} 2 & 3 & 6 & 4 \\ 3 & 5 & 5 & 5 \\ 3 & 7 & 3 & 6 \end{pmatrix}$$

Se van tomando los instantes de tiempo (filas de `POS_all`) que marca la simulación, es decir, el tamaño del intervalo que se haya establecido, y se realiza el resto de las partes de la simulación con normalidad (ver figura 2.6).

2.3.3. Integración de modelos de movilidad

Es importante explicar cómo se ha llevado a cabo la integración de los modelos en el simulador, posibilitando de esta manera, posibles ampliaciones en futuras versiones.

Existe una integración profunda de los modelos Random Wapoint modificado y Random Direction modificado en la que la gestión del movimiento transcurre al mismo tiempo que el resto de la simulación. En cada intervalo de tiempo se reubican los nodos por el escenario y se ejecutan las partes correspondientes del bloque dos y tres (explicados en posteriores capítulos). Se trata de una integración compleja que obliga a un entendimiento previo del código. Sin embargo, permite un mayor control de la situación y la posibilidad de añadir criterios de control o toma de decisiones.

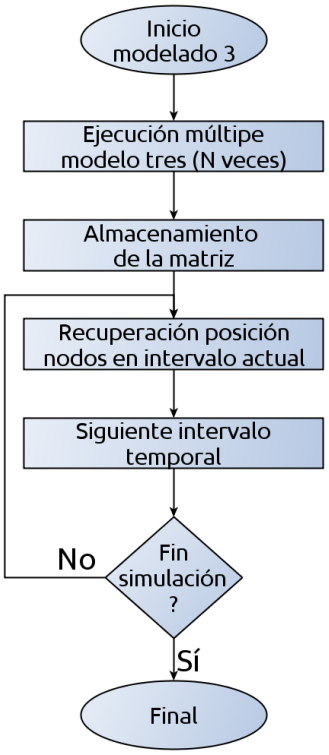


Figura 2.6: Funcionamiento del modelo de movilidad Truncated Levy Walk modificado.

Por otro lado se presenta, como ejemplo de integración superficial, la llevada a cabo con el modelo Truncated Levy Walk modificado. Es la opción propuesta para la integración de nuevos modelos, por su sencillez. La idea principal es generar todo el recorrido que realizan los nodos al inicio de la simulación, mediante la llamada a una función. De manera que quede aislado y su traza de movimiento sea independiente del resto del simulador. Esta opción es más sencilla de implementar, aunque pierde la parte de control que ofrecía la propuesta anterior.

2.4. Evaluación y Conclusiones

La posibilidad de ajustar el diseño de un modelo para simular un entorno real es muy interesante. Este proyecto no pretende hacer hincapié sobre cuales son mejores que otros, sino ofrecer una herramienta de estudio de modelos de movilidad. Se han escogido para este trabajo unos modelos con diferentes comportamientos y características

para ofrecer una visión práctica de los mismos.

A lo largo del capítulo se han ido mostrando las diferencias que presentan unos modelos de movilidad con otros. Éstas vienen dadas por los distintos algoritmos que implementan, los parámetros de configuración de los nodos o las características propias del escenario. Un modelo puede ofrecer comportamientos muy diferentes gracias a las posibilidades que proporciona una configuración basada en un elevado número de parámetros, como pueden ser, velocidades, tiempos de pausa, distancias a recorrer, rebotes, etc. Todo ello hace que se pueda utilizar un único modelo para diversas situaciones.

El comportamiento general de los nodos en cada uno de los modelos implementados se muestra en la figura 2.7 en la que se presenta un ejemplo de tres recorridos correspondientes a cada uno de los modelos en una simulación de 5 horas.

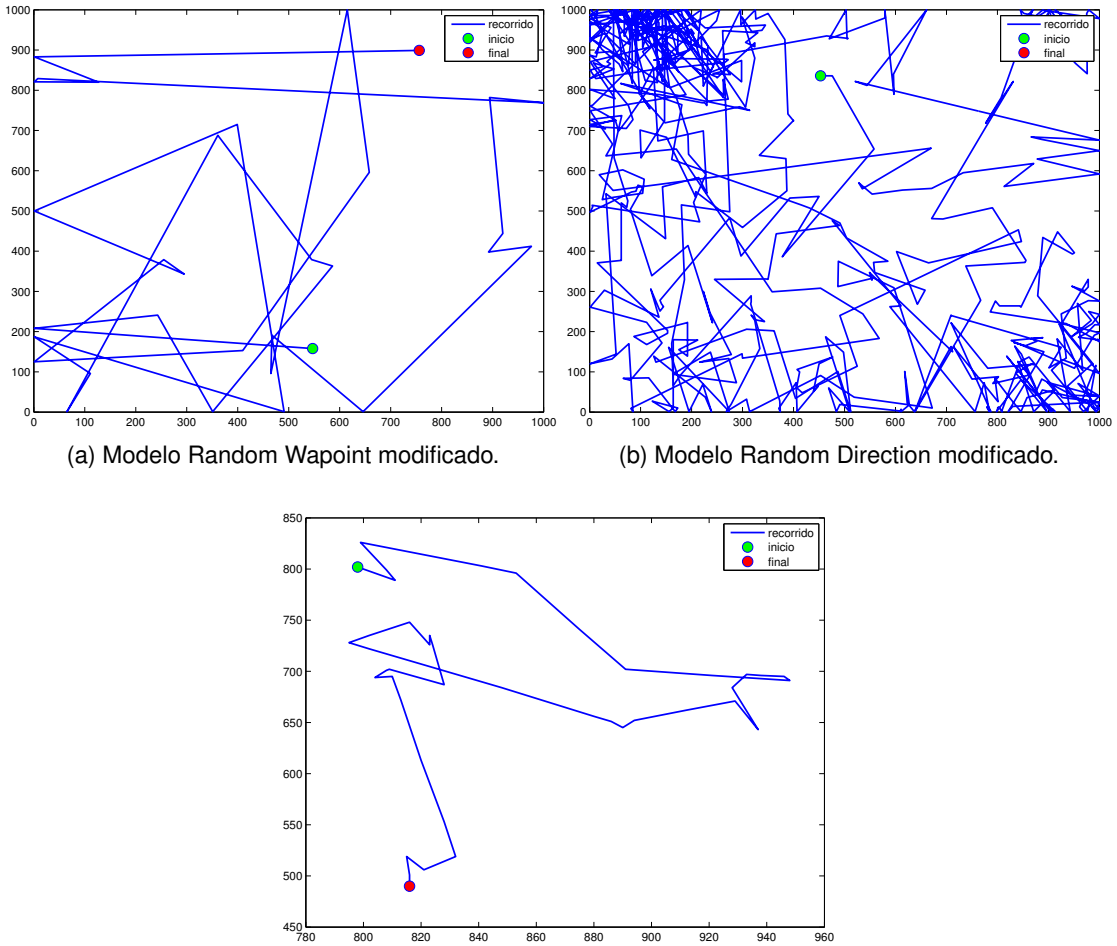
Se trata de unas gráficas que representan el escenario sobre el que se encuentran los nodos. De forma que ambos ejes vienen expresados en metros.

La figura 2.7a corresponde a un nodo movido mediante el modelo Random Wapoint modificado, tratándose de un movimiento aleatorio que genera bastantes tramos rectilíneos. En la figura 2.7b, el movimiento del nodo es mucho más caótico debido a que se encuentra modelado por el modelo Random Direction modificado. Conviene recordar que generaba tiempos internos dentro de cada intervalo de la simulación dotando al nodo de un movimiento mucho más complejo.

Este modelo presenta el problema de retenciones en las esquinas del escenario, que provoca que los nodos puedan quedar parcialmente atrapados en estas zonas debido a la configuración de los rebotes. Es destacable también que el recorrido del nodo del modelo Random Direction modificado es mucho mayor que en el anterior caso.

Por último, con el modelo Truncated Levy Walk modificado se ha generado el movimiento del nodo de la figura 2.7c. Es un movimiento más controlado, en el que a desplazamientos largos el nodo se mueve en línea recta, mientras que en los cortos el movimiento está centrado en la zona en la que se encuentra. Para entender la particularidad advertida, en la figura 2.8 se muestra un recorrido de un nodo simulado con el modelo Truncated Levy Walk modificado, configurado con velocidades entre 1Km/h y 3Km/h y 10 horas de duración. De esta forma se puede apreciar lo dicho respecto a la movilidad de los nodos frente a distancias largas y cortas.

Para el escenario donde se desarrollará la simulación de este proyecto, se han di-



(a) Modelo Random Wapoint modificado.

(b) Modelo Random Direction modificado.

(c) Modelo Truncated Levy Walk modificado.

Figura 2.7: Recorridos de los nodos en una simulación de 5 horas utilizando los tres modelos implementados (a), (b) y (c).

señado dos recintos de prueba con características diferentes: uno cerrado, configurable en anchura y longitud; y un escenario tipo avenida comercial, con ralentización de nodos en las zonas superior e inferior (simulando la zona de tiendas) y configuración de pasillo para que los nodos entren y salgan de la avenida por los laterales (ver figura 2.9). De forma arbitraria se ha escogido el tamaño de la zona comercial y el tiempo de parada en estas. En ambos casos no son valores fijos, sino que se toman proporcionalmente a partir del tamaño del escenario y del tiempo de pausa marcado en los parámetros iniciales.

La disposición de los nodos en el escenario, cómo se van distribuyendo, zonas en las que pueda haber una alta densidad o detección de problemas y/o anomalías en el modelo,

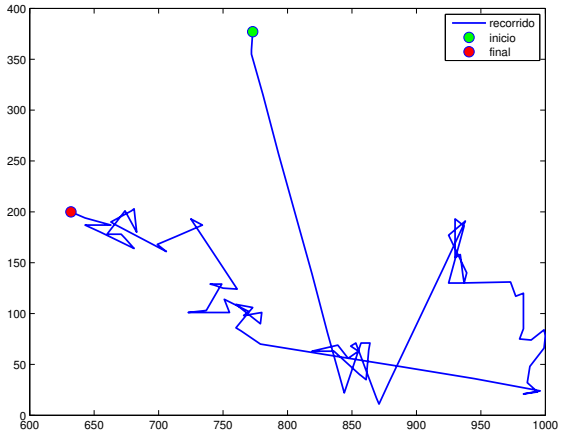


Figura 2.8: Recorrido de un nodo mediante el modelo de movilidad Truncated Levy Walk modificado.

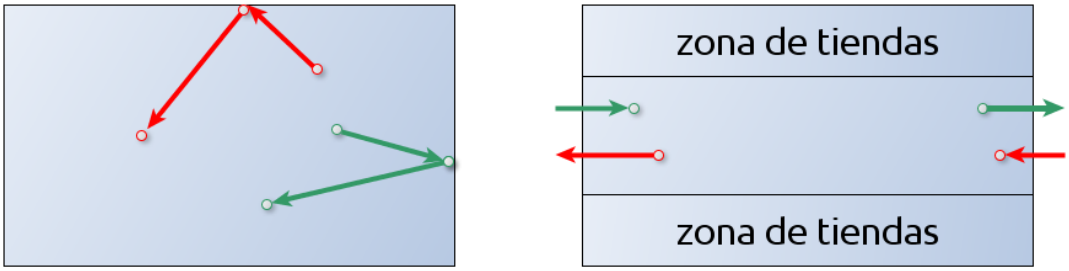
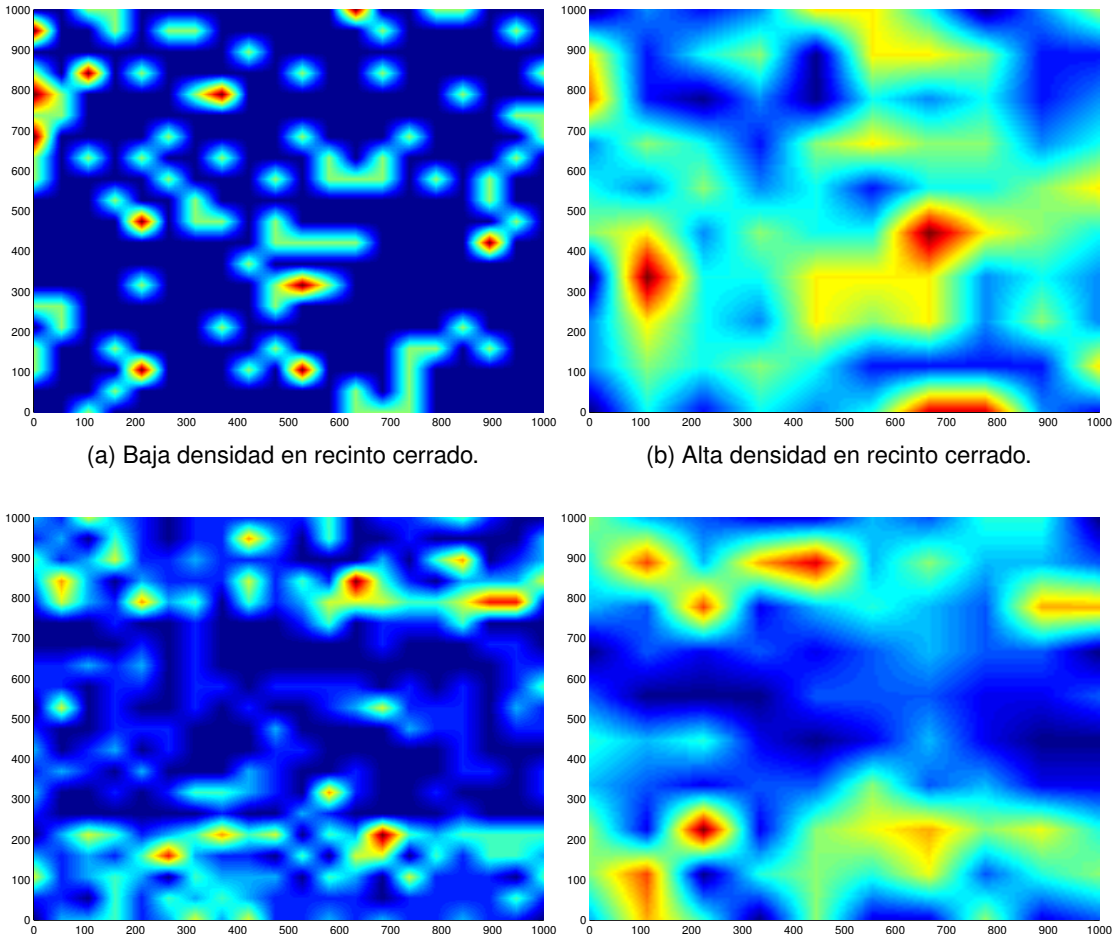


Figura 2.9: (A) Escenario cerrado en el que los nodos rebotan al llegar a cualquier pared. (B) Escenario de tipo avenida comercial.

son características de una gran importancia que hacen necesario alguna medición que permita analizar esas situaciones.

En la figura 2.10 se muestran cuatro instantes de simulaciones diferentes en las que se observa cómo están repartidos los nodos. En las dos imágenes superiores (figuras 2.10a y 2.10b) correspondientes a dos simulaciones con el escenario cerrado se observa la dispersión de estos. Las figuras 2.10c y 2.10d corresponden con otros dos instantes de tiempo para simulaciones con distinto número de nodos, pero sobre el escenario de tipo avenida comercial.

Tomar varias instantáneas de este estilo de una misma simulación para poder ir vien-



(a) Baja densidad en recinto cerrado.

(b) Alta densidad en recinto cerrado.

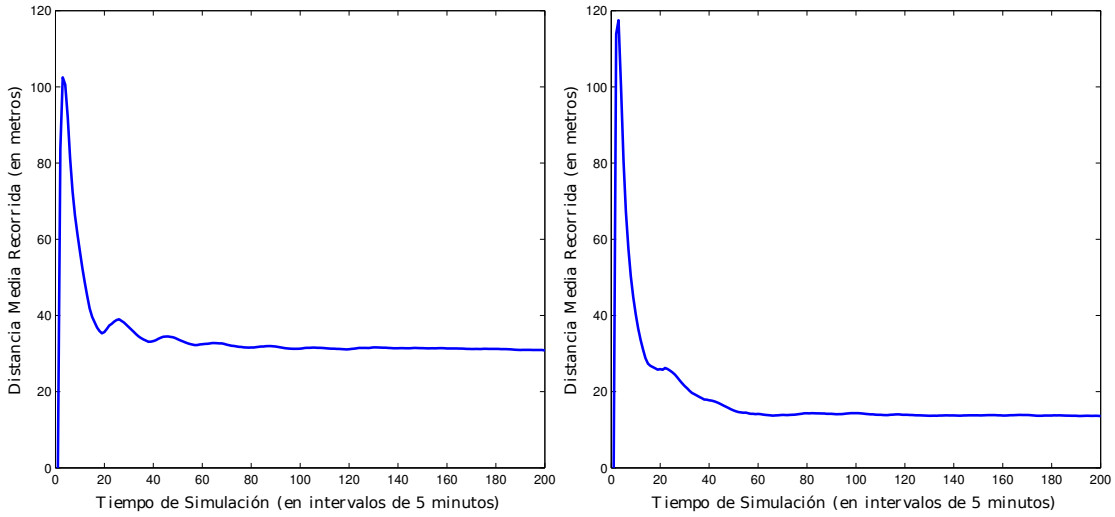
(c) Baja densidad en recinto comercial.

(d) Alta densidad en recinto comercial.

Figura 2.10: Densidad de los nodos en los dos escenarios, en una superficie $500m^2$. (a) y (b) en un recinto cerrado, (c) y (d), en un recinto comercial.

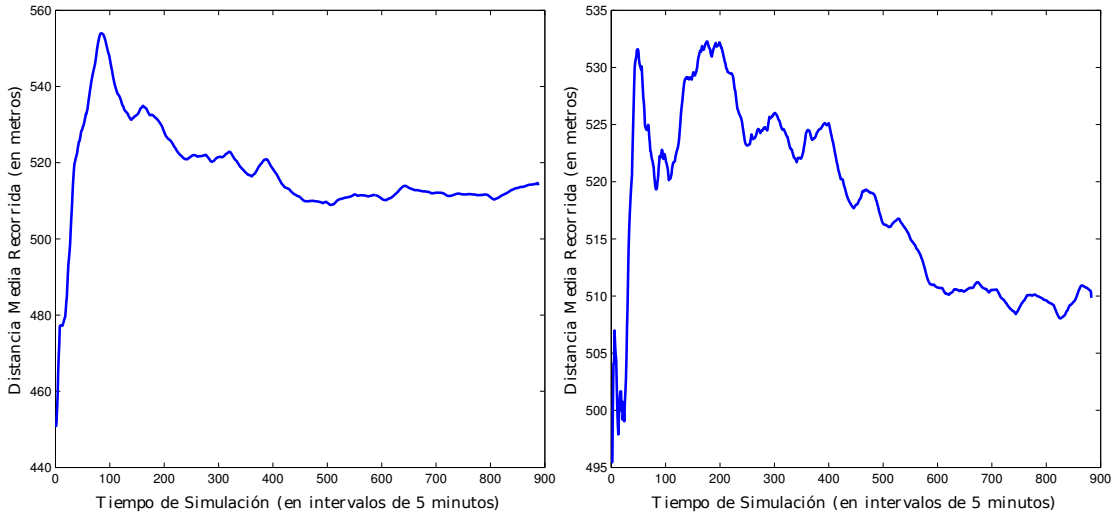
do la evolución de un modelo puede resultar de gran utilidad, tanto si es de alguno ya integrado, o sobre todo si se trata de un modelo nuevo. Es importante saber si estamos ante un modelo estable, es decir, que se mantendrá en el tiempo sin necesidad de alterar su comportamiento, o inestable, haciendo que a medida que la simulación avance en el tiempo, haga que los nodos vayan quedándose en los bordes del recinto o terminen todos en alguno de los dos estados (pausados o en movimiento).

Por último, el simulador ofrece unas estadísticas del comportamiento de los nodos, con diferentes parámetros de configuración, para realizar un estudio del funcionamiento del bloque. Un ejemplo de la utilidad que representa esta prestación se expone en el



(a) Recinto cerrado, modelo Random Wapoint modificado.

(b) Recinto comercial, modelo Random Wapoint modificado.



(c) Recinto cerrado, modelo Random Direction modificado.

(d) Recinto comercial, modelos Random Direction modificado.

Figura 2.11: Distancia media recorrida en un intervalo de tiempo en cuatro simulaciones diferentes. (a) y (b) usando Random Waypoint modificado, (c) y (d) usando Random Direction modificado.

conjunto de gráficas de la figura 2.11. En ellas se observa la distancia media recorrida en un intervalo de tiempo (de izquierda a derecha y de arriba a abajo) en una superficie de $500m^2$:

- **1.** Simulaciones de 0 a 1000 minutos a intervalos de 5 minutos. Analizando el comportamiento de 300 nodos (modelados con el modelo Random Wapoint modificado). En el escenario cerrado.
- **2.** Simulaciones de 0 a 1000 minutos a intervalos de 5 minutos. Analizando el comportamiento de 300 nodos (modelados con el modelo Random Wapoint modificado). En el escenario comercial.
- **3.** Simulaciones de 0 a 4500 minutos a intervalos de 5 minutos. Analizando el comportamiento de 300 nodos (modelados con el modelo Random Direction modificado). En el escenario cerrado.
- **4.** Simulaciones de 0 a 4500 minutos a intervalos de 5 minutos. Analizando el comportamiento de 300 nodos (modelados con el modelo Random Direction modificado). En el escenario comercial.

La información de la gráfica viene dada en metros e intervalos de tiempo de minutos, de forma que el eje X representa el tiempo total de la simulación llevada a cabo y el eje Y la distancia en media recorrida en metros.

Comparando los dos escenarios de prueba implementados para validar el funcionamiento del modelo Random Wapoint modificado (ver figuras 2.11a y 2.11b), se observa cómo en ambos casos existe un transitorio de estabilización de la distancia media debido a la configuración inicial de arranque con la que parten los nodos. Para el ejemplo del escenario cerrado este transitorio dura 200 minutos, prácticamente igual ocurre en las simulaciones realizadas en el escenario de la avenida comercial. El punto de estabilización en la primera gráfica se encuentra en torno a los 40 metros, puede estimarse que la distancia que recorren los nodos (teniendo en cuenta a los parados y los que están en movimiento) en media tiende a esa. Como era de esperar en el escenario comercial éste valor disminuye hasta los 20 metros, debido a la implementación de las zonas de pausa (zonas de tiendas) descritas anteriormente. Correrá a cargo del diseñador decidir en base a los datos obtenidos si el modelo que utiliza es adecuado o no para el entorno que esté estudiando.

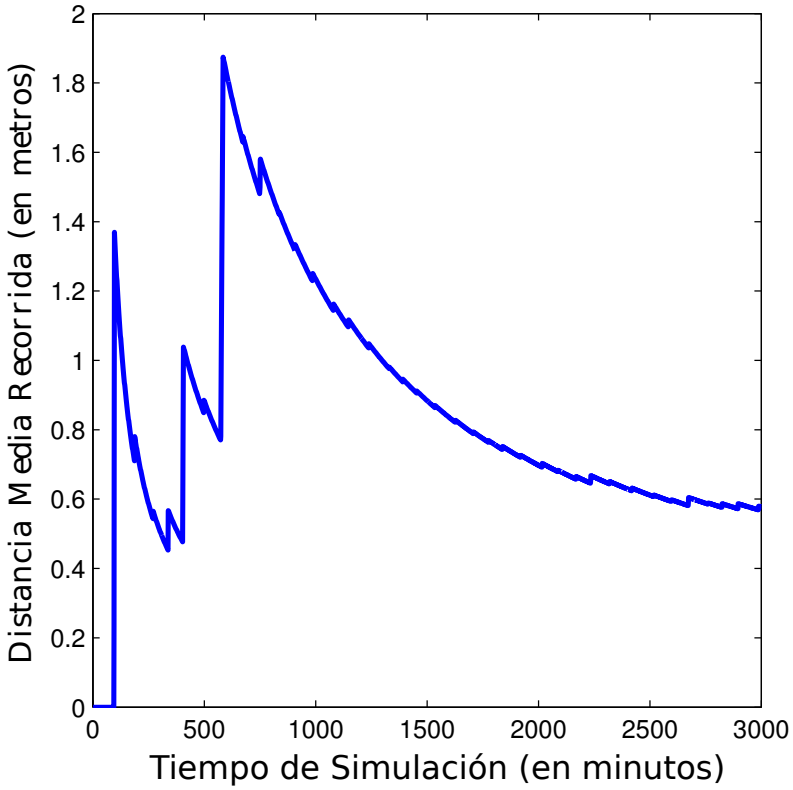


Figura 2.12: Distancia media en una simulación con el modelo Truncated Levy Walk modificado.

Las otras dos gráficas (ver figura 2.11c y 2.11d) corresponden a la comparativa de los dos escenarios implementados, modelando el movimiento de los nodos mediante el modelo Random Direction modificado. Se observa a primera vista las diferencias que presenta este modelo con respecto al anterior, tratándose de un movimiento mucho más caótico y cuya media converge más lenta. Este modelo utiliza un mecanismo pseudoaleatorio para regular el número de nodos parados, de forma que el transitorio de estabilización no es significativo, además la gráfica presenta pequeños picos. Sin embargo, interesa darse cuenta de la diferencia de distancia (en media) recorrida respecto al otro modelo, ya que en ambos casos no baja de 500 metros. Se trata de un resultado razonable, ya que al simular sobre el escenario cerrado (no tiene zonas de retención de nodos) e incorporar el mecanismo de parado (mantiene un porcentaje de nodos en movimiento), hace que la media esté acorde con la introducida por parámetro (entre 1 y 1000m).

Respecto a las gráficas de distancia, se ha realizado una simulación que detalla el

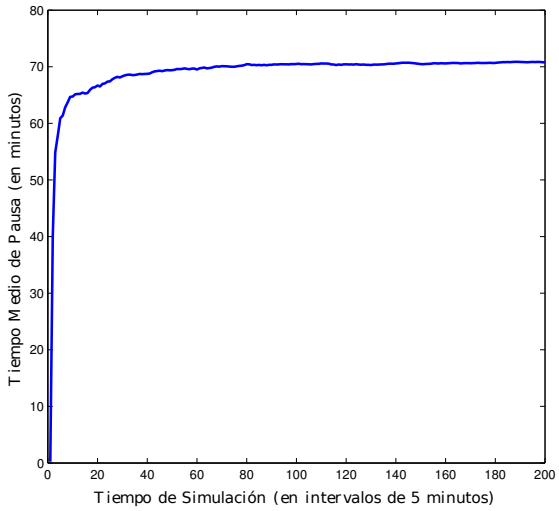
comportamiento interesante del modelo Truncated Levy Walk modificado (ver figura 2.12). Los parámetros de la simulación son los siguientes:

- Tiempo de simulación: 3000 minutos.
- Escenario de prueba: recinto cerrado.
- Número de nodos: 100.
- Tiempos de pausa: entre 60 y 100 minutos (se escoge un tiempo muy elevado para poder ver el detalle del patrón de movimiento).

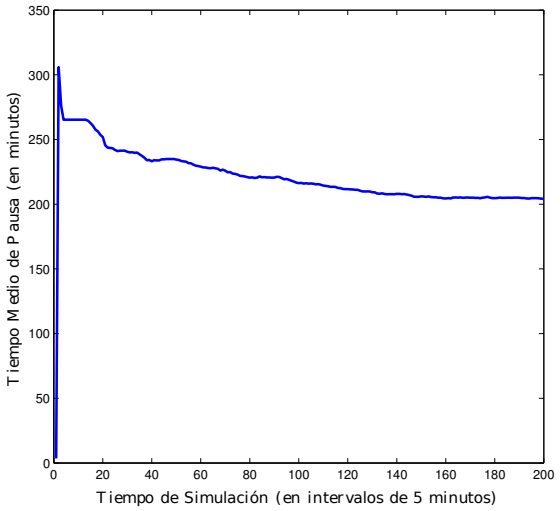
Se observa cómo al tratarse de un modelado que utiliza un algoritmo de movimiento basado en leyes de potencias (vuelos de Lèvy), la distancia media que recorren los nodos sufre una serie de picos y caídas suaves que representan los trayectos largos en instantes de tiempo más concretos (picos), es decir, son cambios de posición rápidos de los nodos y las agrupaciones posteriores en la zona donde quede el nodo (caídas suaves). La distancia media recorrida es muy baja ya que se utilizaron tiempos de pausa muy altos en una simulación de larga duración para poder observar el comportamiento de este modelo.

A continuación se presenta otro conjunto de gráficas (ver figura 2.13) correspondiente al tiempo de pausa medio de los nodos parados. Las simulaciones llevadas a cabo con las siguientes (de izquierda a derecha y de arriba a abajo):

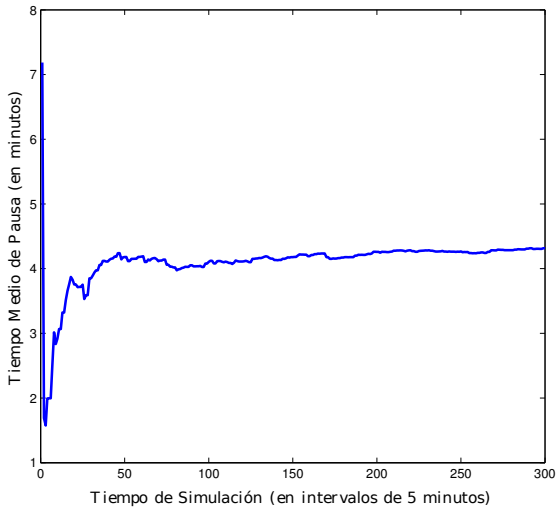
- **1.** Simulaciones de 0 a 1000 minutos a intervalos de 5 minutos. Analizando el comportamiento de 300 nodos (modelados con el modelo Random Wapoint modificado) a velocidades de entre 1Km/h y 3Km/h. En el escenario cerrado.
- **2.** Simulaciones de 0 a 1000 minutos a intervalos de 5 minutos. Analizando el comportamiento de 300 nodos (modelados con el modelo Random Wapoint modificado) a velocidades de entre 1Km/h y 3Km/h. En el escenario comercial.
- **3.** Simulaciones de 0 a 1500 minutos a intervalos de 5 minutos. Analizando el comportamiento de 300 nodos (modelados con el modelo Random Direction modificado) a velocidades de entre 1Km/h y 3Km/h. En el escenario cerrado.
- **4.** Simulaciones de 0 a 1500 minutos a intervalos de 5 minutos. Analizando el comportamiento de 300 nodos (modelados con el modelo Random Direction modificado) a velocidades de entre 1Km/h y 3Km/h. En el escenario comercial.



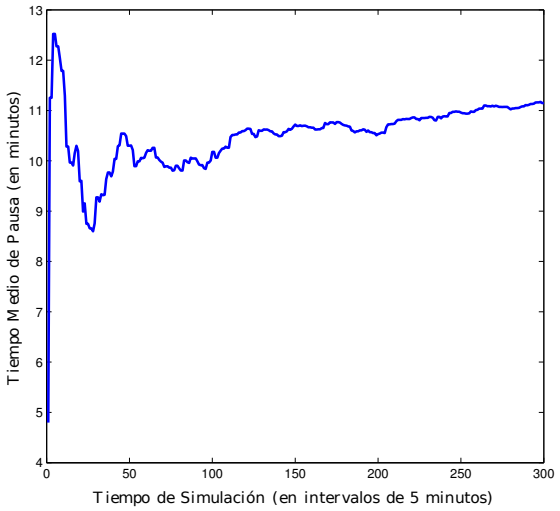
(a) Recinto cerrado, modelo Random Wapoint modificado.



(b) Recinto comercial, modelo Random Wapoint modificado.



(c) Recinto cerrado, modelo Random Direction modificado.



(d) Recinto comercial, modelos Random Direction modificado.

Figura 2.13: Tiempo de pausa medio en cuatro simulaciones con distintas configuraciones.

La información de las gráficas (ver figura 2.13) viene dada en metros e intervalos de tiempo. El eje X representa el tiempo total de la simulación, dado en intervalos de tiempo y el eje Y corresponde al tiempo medio de pausa de los nodos, es decir, durante cuanto tiempo han estado parado los nodos a lo largo de un intervalo.

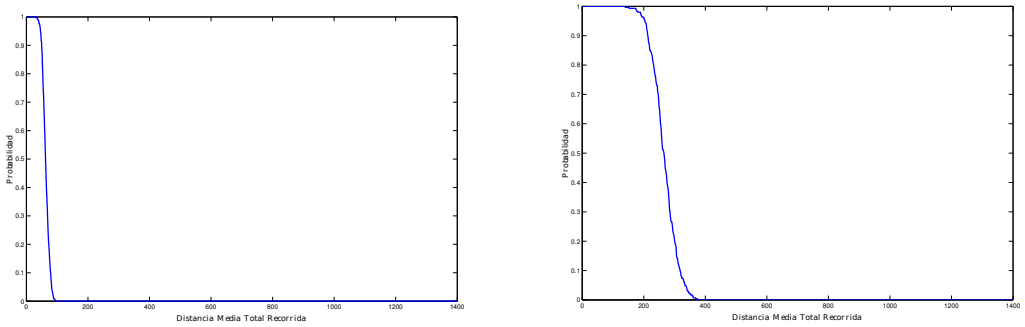
En las figuras 2.13a y 2.13b se presenta una situación especial a tener en cuenta por el diseñador a la hora de analizar la información que ofrece esta gráfica. Para esos dos conjuntos de simulaciones se han escogido unos tiempos de pausa elevados (entre 60 y 100 minutos), que hace que en las simulaciones inferiores a esos tiempos no den una medida real del tiempo de pausa, ya que dicha media es mayor que el tiempo de simulación. Eso es debido a que un nodo selecciona un tiempo de pausa independientemente del tiempo de simulación.

Es destacable el diferente comportamiento que presentan los nodos en función del escenario en el que se encuentren, es decir, para el escenario cerrado los nodos describen un crecimiento en el tiempo de pausa hasta estabilizarse, mientras que utilizando el recinto comercial, el conjunto de los nodos va reduciendo el tiempo de pausa hasta alcanzar el valor medio del total. Esto es debido a la configuración inicial con la que empieza el movimiento de los nodos.

Por otro lado, el par de gráficas inferior (ver figuras 2.13c y 2.13d), que corresponde con el conjunto de pruebas relativo al modelo Random Direction modificado, presentan un incremento paulatino en el tiempo de pausa medio que se estabiliza a los 250 minutos aproximadamente, tras el cual se observa cómo los nodos pertenecientes al escenario comercial poseen un tiempo de pausa mayor que en el recinto cerrado. Estos tiempos de pausa son acordes con los introducidos por parámetro ya que estaban configurados para tiempos entre 1 y 10 minutos. La figura 2.13d presenta una media superior a la introducida por parámetro ya que al tratarse de una simulación en el escenario comercial existen condiciones de entorno que modifican el comportamiento de los nodos.

A continuación se presenta el último conjunto de gráficas 2.14, en las que la información de cada una de ellas viene dada por un eje X que determina la distancia en metros máxima recorrida por los nodos durante un intervalo de tiempo y en el eje Y se representa el porcentaje de nodos que alcanzan esa distancia.

Se han realizado dos simulaciones para interpretar y entender el valor de dicha gráfica. En la figura 2.14 se muestra la distancia máxima que alcanzaron los nodos y el porcentaje de ellos que lo hicieron. La figura 2.14a corresponde con una simulación de 300 nodos



(a) Modelo Random Wapoint modificado durante 1000 minutos.

(b) Modelo Random Direction modificado durante 1000 minutos.

Figura 2.14: Porcentaje de nodos en función de la distancia, utilizando dos modelos diferentes.

durante 1000 minutos utilizando el modelo de movilidad Random Waypoint modificado. Se observa un comportamiento similar en todos los nodos puesto que recorren aproximadamente 100 metros. Respecto a la figura 2.14b, se trata de una simulación con las mismas características que la anterior salvo el modelo utilizado, que en este caso usa el Random Direction modificado. La distancia media total recorrida mediante este modelo es superior al anterior, llegando a doblarlo. Es un comportamiento previsible acorde con las propiedades de cada uno de los modelos.

Los valores de la distancia resultante pueden parecer bajos, esto es debido a la configuración escogida para los tiempos de pausa, se trata por tanto de criterios arbitrarios de simulación que deberá considerar el diseñador.

La figura 2.15 corresponde a tres simulaciones de 3000 minutos de duración y tiempos de pausa estándares entre 1 y 10 minutos. Presenta una comparativa entre los tres modelos de movilidad implementados. Se observan las diferencias respecto a la distancia total recorrida en un intervalo de tiempo por los nodos, observando que el modelo que menos movimiento imprime a los nodos es el Truncated Levy Walk modificado, en el que los nodos recorren un total de 400 metros, por contra el Radom Direction modificado es el que hace que los nodos lleven más movimiento que los otros tres.

Por último, se realiza una simulación de un escenario con 100 nodos, utilizando el modelos Truncated Levy Walk modificado, durante 3000 minutos. La figura 2.16 presenta de forma conjunta dos de las gráficas ya explicadas anteriormente (distancia media y pausa media). En la primera gráfica se aprecia que tras un tiempo de 500 minutos la

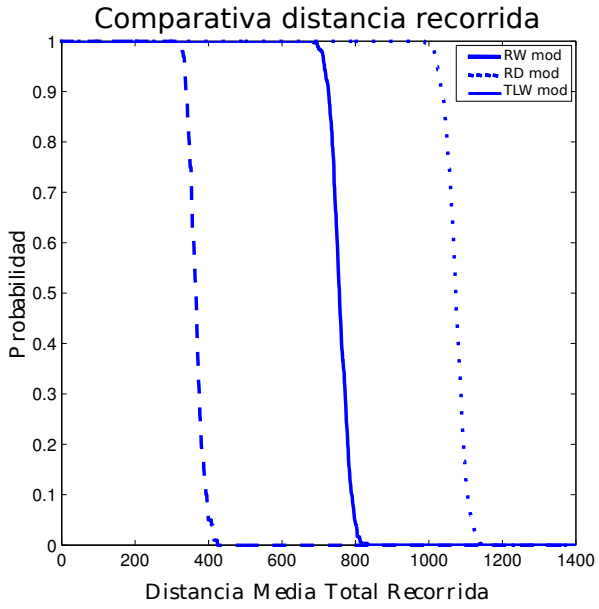
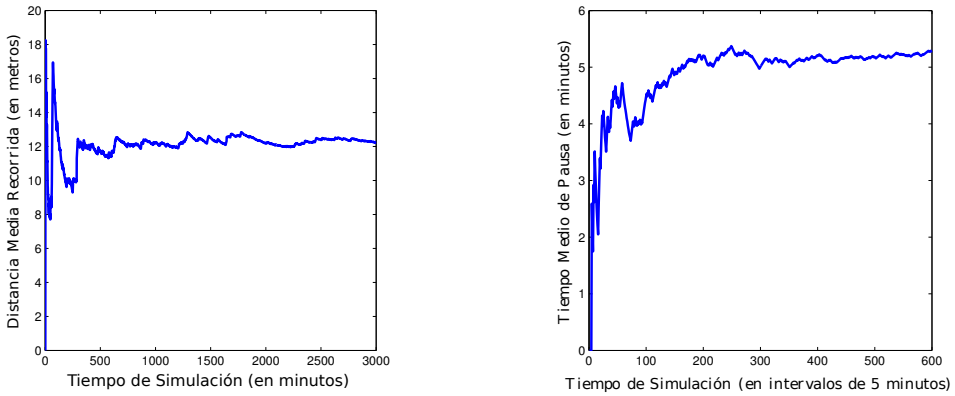


Figura 2.15: Porcentaje de nodos en función de la distancia en una simulación con modelo Random Waypoint modificado, Random Direction modificado y Truncated Levy Walk modificado durante 3000 minutos.

distancia que recorren los nodos en media se estabiliza en torno a unos 12 metros. El tiempo de pausa que toman los nodos en media se estabiliza transcurridos 1000 minutos de la simulación, para centrarse el tiempo medio de pausa en 5 minutos, esto es coherente con los parámetros de configuración establecidos, los cuales fijaban el tiempo de pausa mínimo y máximo en 1 y 10 minutos respectivamente.



(a) Distancia media por intervalo

(b) Tiempo de pausa medio por intervalo

Figura 2.16: Distancia y pausa medios en una simulación con el modelo Truncated Levy Walk modificado.

Capítulo 3

Encaminamiento en redes ad hoc

Este capítulo explica el segundo bloque principal del proyecto, responsable de la parte de encaminamiento de los nodos. En él se verá las características especiales que existen dentro de este tipo de redes.

3.1. Introducción

El tipo de redes sobre las que se centra este proyecto (redes ad hoc) hacen del encaminamiento de paquetes entre diferentes nodos una de las partes más críticas del conjunto, ya que al tratarse de redes móviles van a cambiar de topología con cierta frecuencia debido al movimiento aleatorio de los nodos por el escenario.

Las principales características de las redes ad hoc a tener en cuenta a la hora de realizar el encaminamiento son las siguientes [9]:

- Terminales independientes: cada terminal o nodo puede asumir el papel de emisor, receptor o nodo intermedio.
- Conexiones inalámbricas: al no existir ningún tipo de infraestructura fija, los terminales usan el aire como canal de comunicación.
- Operación distribuida: no existe ningún elemento central que se encargue de la gestión y control de la red. Todos los nodos son iguales y la gestión está repartida por igual entre ellos.

- Topología dinámica de red: debido a la inexistencia de una infraestructura fija y a la movilidad de los nodos, la topología de la red puede ser altamente cambiante. Las redes deben adaptarse rápidamente a los cambios de tráfico generado por los nodos, a los distintos patrones de movimientos y a las condiciones de propagación.
- Capacidad variable de los enlaces: al tratarse de un medio de transmisión compartido, el canal de transmisión cambia constantemente los niveles de ruido, atenuación e interferencias. Además, en una transmisión extremo a extremo pueden participar varios enlaces distintos y la ruta puede cambiar varias veces en una misma transmisión.
- Dispositivos de baja potencia: el consumo de energía se debe reducir lo máximo posible ya que los nodos son móviles y funcionan con baterías de vida limitada.

El encaminamiento tiene como objetivo encontrar rutas entre los nodos en relación con uno o varios parámetros de coste. Los modos de encaminamiento tradicionales, protocolos y algoritmos, tras la aparición de este tipo de redes, han tenido que evolucionar debido a las diferencias respecto a las redes fijas.

En este capítulo se realiza un estudio de los diferentes algoritmos y protocolos de encaminamiento, para sentar las bases sobre las que se apoya este bloque.

A nivel práctico se implementan dos técnicas de selección de enlaces que están incluidas en el simulador, y se presenta el algoritmo de encaminamiento utilizado para este.

3.2. Estudio del Estado del Arte

Antes de pasar a describir los diferentes enfoques y principales protocolos de encaminamiento se deben tener en cuenta los siguientes criterios para una red ad hoc [10]:

- **Minimización de costes de control:** reducción del numero y tamaño de los mensajes intercambiados para temas de control; ya que provocan un consumo de ancho de banda que las redes de este tipo no se pueden permitir desperdiciar debido a sus limitados recursos.
- **Minimización de costes de procesamiento:** la complejidad de los protocolos es un factor a tener en cuenta ya que el diseño de algoritmos de encaminamiento

complejos provoca un mayor consumo en los nodos y al tratarse de dispositivos móviles, esto pasa a ser un asunto prioritario.

- **Capacidad multisalto:** el protocolo ha de ser capaz de encontrar rutas a los nodos de varios saltos ya que el rango de transmisión de los nodos en este tipo de redes es limitado, haciendo que el destino no este dentro del rango de alcance del origen.
- **Mantenimiento dinámico de la topología:** es necesario mantener un camino mientras los nodos intermedios se mueven, incluso si lo hiciesen el origen y/o el destino, para mantener una comunicación entre nodos finales. Debido a la naturaleza cambiante de las redes ad hoc, una ruta establecida entre dos nodos es probable que pierda conectividad entre alguno de los enlaces que la componen en algún momento.
- **Prevención de bucles:** un bucle se produce cuando un nodo utiliza como siguiente salto a otro que ya se encontraba en el camino de esa ruta. Esto provoca un consumo de ancho de banda y de capacidad de procesamiento que no se debe tolerar debido a la capacidad limitada que suelen tener los nodos que conforman estas redes.

En base a lo expuesto se han desarrollado diferentes protocolos de encaminamiento con distintas características y funcionalidades. Para resumir estas características se ha escogido una propuesta de clasificación [11] [12] presentada en la figura 3.1.

La clasificación presenta tres niveles. El primer nivel distingue entre aquellos que son uniformes y los que no lo son. Los primeros (**uniformes**), corresponden a un comportamiento en el que todos los nodos de la red desempeñan el mismo rol, poseyendo las mismas funciones y características. Como ventaja presentan la ausencia de coste de mantenimiento de la estructura de la red, sin embargo son muy poco escalables. Al otro lado están los protocolos (**no uniformes**) con una estructura jerárquica, de manera que alguno de los nodos pertenecientes a la red asumirá algún rol especial. En algunos casos se los podrá dotar de una mayor capacidad para permitir a la red desplegar algoritmos más complejos y realizar una mejor gestión de la red, pues por norma general se tendrá un mayor conocimiento de lo que está pasando en esta. Presentan la ventaja de ser más escalables con la contrapartida del coste que provoca el mantenimiento de la estructura.

El siguiente nivel ilustra los procedimientos que puede seguir un protocolo a la hora de realizar el descubrimiento de caminos. De tipo **proactivo** implica un funcionamiento

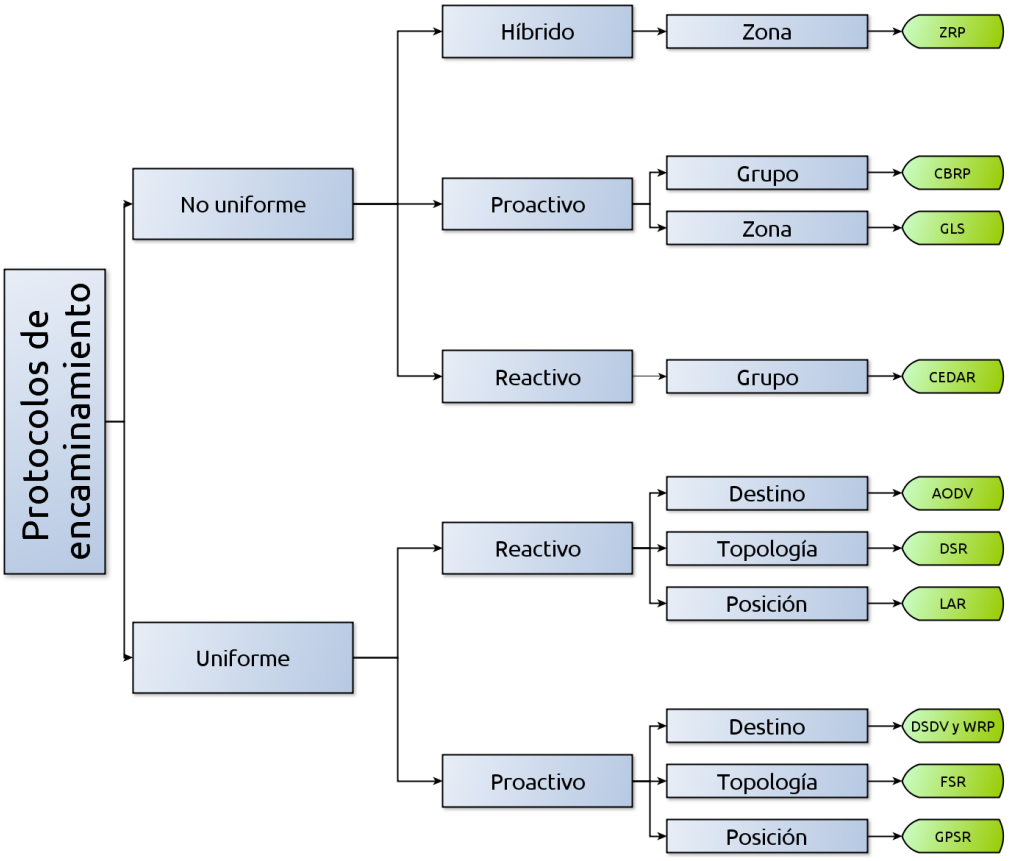


Figura 3.1: Clasificación de los protocolos de encaminamiento.

basado en tablas que almacenan información referente a las rutas de un nodo con todos los demás de la red. Para el mantenimiento de esta información se enviarán mensajes de forma periódica o por cambios en la red. **Reactivos**, aquellos en los que las rutas de encaminamiento sólo son buscadas cuando se necesita establecer una comunicación. Un nodo que desee enviar datos a otro inicia un proceso de descubrimiento de la ruta que finaliza cuando se encuentra la ruta o si tras probar todos posibles caminos entre los nodos, no se encuentra ninguno al destino. Una ventaja del funcionamiento reactivo es el menor gasto de control de mantenimiento, además de una mayor escalabilidad. Sin embargo, los retardos que puede sufrir un nodo que implemente un protocolo reactivo son mayores que en el otro caso. Y por último, están los protocolos **híbrido** que combinan los dos anteriores, normalmente los protocolos de encaminamiento híbrido para redes móviles explotan las arquitecturas de red jerárquica, por lo que suelen ser protocolos no uniformes.

El tercer nivel representa un criterio de clasificación distinto para los protocolos uniformes que para los no uniformes. En el caso de los primeros, indica el tipo de información que manejan los nodos para el encaminamiento. Esta puede ser **topológica**, manejando información global de la red: son protocolos basados en estado de enlaces, enviando a toda la red la lista de los vecinos con los que tienen establecida una conexión. Otra opción es manejar información referente al **destino**, es decir, vector distancias: los nodos mandan a sus vecinos la distancia a la que se encuentran de otros nodos). También pueden manejar información de **posición**, en la que cada nodo conocerá las coordenadas en las que se encuentra y la de los demás. La idea es que un nodo, en base a este conocimiento, encamine sus mensajes al destino por el siguiente salto más cercano a éste, reduciendo el número de saltos.

El criterio de clasificación del tercer nivel de los protocolos no uniformes es en función de la organización, que puede ser por **grupo**, haciendo que los nodos se vayan agrupando en torno a unos nodos específicos que actuarán como principales, o por **zona**, refiriéndose a la agrupación por zona geográfica.

3.2.1. Algoritmos de descubrimiento de camino

Algunos de los algoritmos de descubrimiento de camino más usados son [13]:

- **Bellman-Ford:**

  Algoritmo que busca el camino más corto en un grafo dirigido ponderado [13] (en el que el peso de alguna de las aristas puede ser negativo). El algoritmo de Dijkstra resuelve este mismo problema en un tiempo menor, pero requiere que los pesos de las aristas no sean negativos.

  Basado en *vector de distancia*, este algoritmo opera de la siguiente manera: cada nodo mantiene una tabla de encaminamiento que contiene un registro de cada nodo de la red. Esta entrada comprende dos partes: la línea preferida de salida hacia ese destino y una estimación del tiempo o distancia a ese destino. La métrica usada podría ser la cantidad de saltos o el retardo de tiempo en milisegundos. Cada nodo conoce la *distancia* a cada uno de sus vecinos. Estas tablas se actualizan intercambiando información con los vecinos.

- **Dijsktra:**

Este algoritmo, dado un grafo con etiquetas no negativas [13], trata de calcular el coste del camino mínimo desde un vértice dado (nodo) al resto. Por lo tanto, intenta averiguar el camino más corto para llegar a un punto partiendo de otro (determinar la secuencia de aristas para llegar a un nodo a partir del otro con un coste mínimo). Las aristas dan el coste (distancia o similares) de la conexión entre dos puntos. Se define el camino de coste mínimo entre dos nodos como aquel en el que la suma de los pesos de las aristas que lo forman es la más baja entre las de todos los caminos posibles entre ellos.

Se basa en *estado de enlace*, de manera que un nodo comunica a los restantes nodos de la red cuáles son sus vecinos y a qué distancia está de ellos. Con la información que un nodo de la red recibe de todos los demás, puede construir un mapa de la red y sobre él, calcular los caminos óptimos.

Cada nodo se etiqueta con su distancia al nodo de origen a través de la mejor ruta conocida. Inicialmente no se conocen rutas, por lo que todos los nodos tienen la etiqueta infinito. A medida que avanza el algoritmo y se encuentran rutas, las etiquetas pueden cambiar, reflejando mejores rutas.

- **K-Shortest Paths:**

  Se trata de un algoritmo de búsqueda de rutas que obtiene los *K* caminos distintos más cortos sin bucles entre dos vértices (nodos) de un grafo [14], haciendo uso del algoritmo de Dijkstra. En primer lugar busca el camino más corto sin bucles. En segundo lugar, los demás caminos siempre se buscan en base al resto de caminos encontrados.

- **Greedy forwarding:**

  Algoritmo en el que los mensajes que generan los nodos incluyen información sobre la localización del nodo destino [15], que es conocida por todos los nodos de la red. Además, los nodos tienen información geográfica de la posición de algunos nodos de la red, llamados vecinos. Esta información es adquirida mediante un sencillo algoritmo de señalización. Cada nodo envía periódicamente un mensaje (baliza) con su identificador y su posición para indicar a los demás nodos que continúa activo. Para evitar que haya sincronización en el envío de balizas con otros vecinos, la transmisión de cada baliza se realiza de forma aleatoria sobre la mitad del intervalo de tiempo entre balizas.

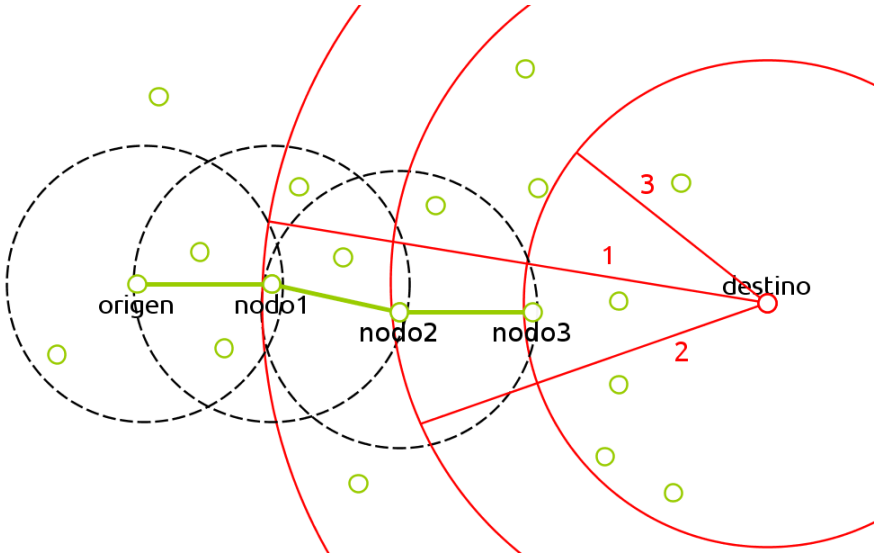


Figura 3.2: Ejemplo de funcionamiento del algoritmo Greedy forwarding.

Cuando un nodo tiene un mensaje que transmitir selecciona de entre sus vecinos al nodo que le proporcione un mayor avance hacia el nodo destino, con la finalidad de minimizar el número de saltos. En la figura 3.2 se ilustra un ejemplo del funcionamiento de este algoritmo. El nodo origen desea enviar datos al nodo destino y éste no se encuentra dentro de su radio de cobertura (línea a rayas). La curva 1 (primera línea roja empezando por la izquierda) con radio igual a la distancia entre destino y *nodo1*, denota que el *nodo1* es el vecino con el que el origen tiene un mayor avance hacia el destino. De igual manera procederán el resto nodos hasta alcanzar al destino.

3.2.2. Protocolos de encaminamiento ad hoc

A continuación se describen las características de los principales protocolos de encaminamiento especialmente diseñados para redes Ad-Hoc (tabla 3.1):

- **DSDV** (Destination Sequence Distance Vector):

  Cada nodo de la red mantiene una tabla de encaminamiento que contiene todos los posibles destinos y el número de saltos a éste. Cada entrada posee un número de

secuencia asignado por el nodo destino. Los números de secuencia permiten distinguir rutas antiguas de rutas modernas. Continuamente se deben enviar mensajes de actualización a través de la red para mantener la consistencia de las tablas. En caso de que haya dos rutas distintas hacia un destino, se usará la que contenga el número de secuencia más moderno. En general, DSDV es un protocolo aceptable en escenarios en los que todos los nodos intervienen en las comunicaciones y en los lo que la movilidad es media.

- **WRP** (The Wireless Routing Protocol):

  Protocolo basado en tablas cuyo objetivo principal es mantener la información actualizada de todos los nodos de la red. Cada nodo es responsable de mantener cuatro tablas:

  - Tabla de distancias.
  - Tabla de encaminamiento.
  - Tabla de coste de ruta.
  - Tabla con la lista de mensajes retransmitidos (MRL).

  La MRL se utiliza para gestionar el envío de los paquetes de actualización de rutas. Los nodos se informan entre ellos de los cambios en las rutas a través de los paquetes de actualización, los cuales son enviados sólo entre vecinos y contienen los elementos a actualizar en las rutas. Los paquetes se envían cuando procesan las actualizaciones recibidas de otros vecinos o cuando ellos mismos detectan un cambio en el enlace con algún vecino. La omisión de mensajes por parte de un nodo ocasionará la ruptura de ese enlace. Cuando un nodo recibe un mensaje de un nuevo nodo, éste nodo será añadido a la tabla de encaminamiento y una copia de esta tabla será enviada al nuevo nodo.

- **AODV** (Ad Hoc On-Demand Distance Vector):

  Evolución de DSDV. Mantiene la idea de los números de secuencia y las tablas de encaminamiento, pero añade el concepto de encaminamiento bajo demanda, es decir, sólo se guarda información de los nodos que intervengan en la transmisión de datos. Las optimizaciones primordiales que consigue en relación a su anterior diseño son el decremento del tiempo de proceso, disminución del gasto de memoria y reducción del tráfico de control por la red.

Protocolo	Tipo	Descubrimiento	Información
DSDV	uniforme	proactivo	destino
WRP	uniforme	proactivo	destino
AODV	uniforme	reactivo	destino
DSR	uniforme	reactivo	topología
ZRP	no uniforme	híbrido	organización en zona

Tabla 3.1: Clasificación de los protocolos de encaminamiento en redes ad hoc.

- **DSR** (The Dynamic Source Routing):

  Algoritmo basado en el concepto de encaminamiento en origen. Los nodos mantienen tablas, cuyas entradas incluyen el destino y la lista de nodos para llegar a él. Las entradas de esta tabla son actualizadas según se aprendan rutas nuevas. El protocolo consta de dos mecanismos: descubrimiento de ruta y mantenimiento de ruta. Cuando un nodo quiere enviar un paquete a un destino, primero consulta su tabla para determinar si dispone de una ruta hacia el destino. Si tiene una ruta válida, la usará para enviar el paquete. Sin embargo, si el nodo no dispone de dicha ruta iniciará un proceso de descubrimiento de ruta.

- **ZRP** (Zone Routing Protocol):

  Protocolo híbrido entre los algoritmos basados en tablas y los basados en encaminamiento bajo demanda. Es utilizado en una clase particular de redes ad hoc llamadas RWNs (Reconfigurable Wireless Networks). Estas redes se caracterizan por tener gran cantidad de nodos, mucha movilidad y alto tráfico.

3.3. Descripción del Bloque de Descubrimiento de Vecinos y Caminos

Esta sección detalla la implementación de la parte de búsqueda de vecino y encaminamiento del proyecto. Encargada de definir las rutas y generar una matriz de adyacencia a partir de las posiciones de los nodos para el posterior tratamiento en el capítulo de optimización. Trata de reflejar la importancia de escoger adecuadamente los parámetros de diseño para que las rutas y caminos entre enlaces puedan darse con normalidad.

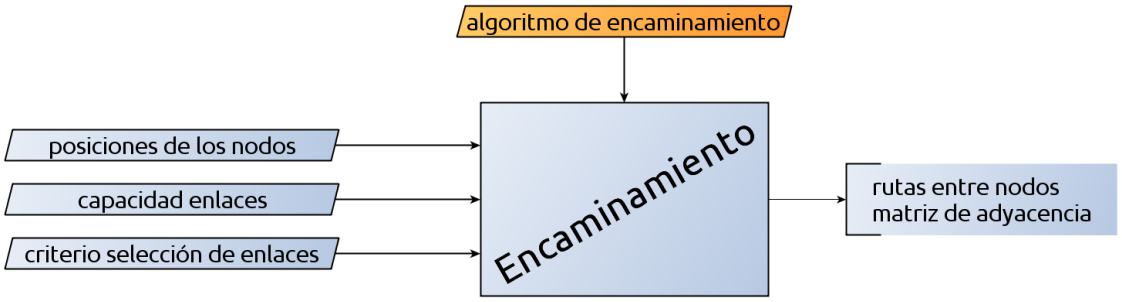


Figura 3.3: Esquema general del bloque de encaminamiento.

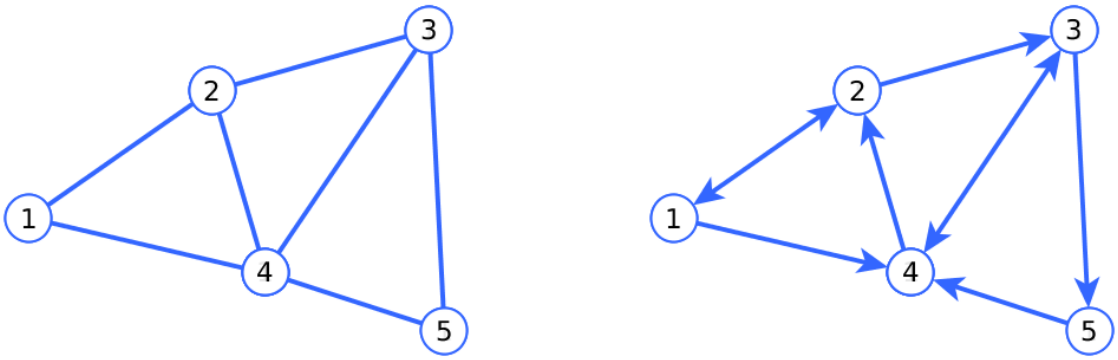


Figura 3.4: Ejemplo red con enlaces bidireccionales y unidireccionales.

El descubrimiento de vecinos se activa a intervalos de tiempo que pueden ir desde el mismo intervalo de tiempo usado para el movimiento de los nodos (como ocurre en la implementación actual del proyecto) hasta otros que hagan menos frecuente el cálculo de rutas.

Las necesidades para realizar el encaminamiento se muestran de manera general en la figura 3.3. Las posiciones en las que se encuentran los nodos vienen dadas por el bloque uno, que en cada intervalo irá actualizando las coordenadas de estos. Los otros dos parámetros son la capacidad o coste de los enlaces y el criterio de selección de enlaces (hace referencia a los vecinos, no a las rutas). Ambos están relacionados hasta el punto en el que la selección de los nodos vecinos o aquellos con los que se va a estar conectado podrán tener definidos criterios que limiten la capacidad, es decir, llevar asociados un coste.

La salida del bloque son las rutas que se establecen entre los nodos que deseen

realizar una comunicación y la matriz de adyacencia, (indica los enlaces que tienen establecidos los nodos con los demás). El número de rutas es una consideración de diseño, siendo siempre superior a uno para que la optimización que se aplica en el siguiente bloque tenga sentido.

Para entender el concepto de la matriz de adyacencia usaremos un ejemplo de una red con cinco nodos con enlaces bidireccionales y unidireccionales (ver figura 3.4). La matrices de adyacencia resultantes serán por lo tanto:

$$Bidireccional \begin{pmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix} \quad Unidireccional \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

La matriz de adyacencia indica qué conexiones están establecidas entre los nodos. Las columnas representan los enlaces de de los nodos con los demás, es decir, la columna uno corresponde a los enlaces que tiene el nodo uno con los demás, y así sucesivamente con el resto de las columnas. Los *"1"* corresponden a un enlace entre el nodo fila y el nodo columna correspondiente. Las redes con enlaces bidireccionales dan como resultado matrices simétricas, considerando la parte superior de la diagonal (excluyendo a esta) se definen todas las conexiones de la red.

Se ha implementado un cambio en la representación de la matriz de adyacencia que es más rica en información ya que en vez de indicar con ceros y unos la ausencia o no de enlaces, el valor de la posición representa el coste de enviar por ese enlace. La matriz resultante, es una matriz de costes, siendo *infinito* los valores que antes eran cero (cuando no hay conexión), y los *"1"* pasan a ser el coste asociado a ese enlace. De esta forma podemos aplicar el algoritmo de encaminamiento de una forma más eficiente.

Antes de poder generar las rutas se define el método de selección de enlaces, para el cual se han implementado dos posibles soluciones (ver figura 3.5):

El primero(figura 3.5A) implementa *anillos de cobertura*, es decir, determina una serie de radios cuyas circunferencias serán los límites de la zonas. Aquellos nodos contenidos en ellas serán con los que el nodo tenga conexión. Configurando varios radios podremos asignarles un determinado coste a cada uno en función del grado de la conexión. El grado de la conexión indica el anillo al que pertenece ese enlace, cuanto mayor sea el grado,

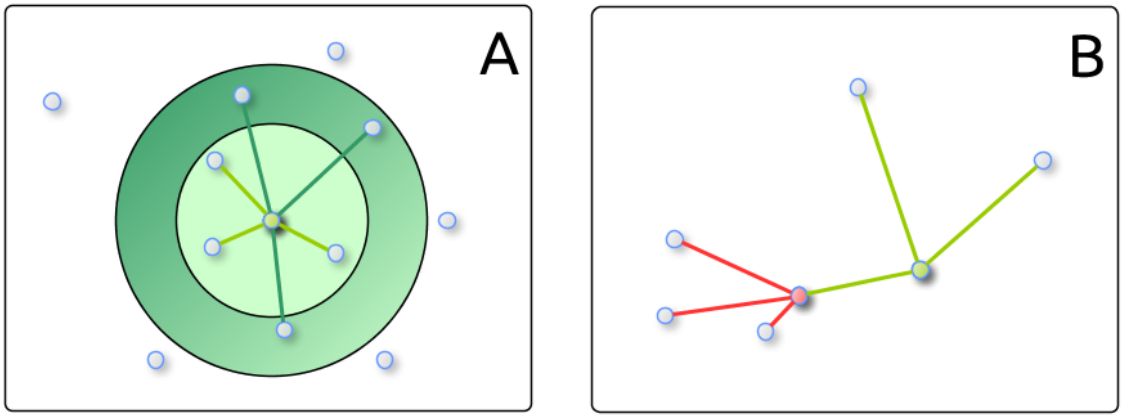


Figura 3.5: Métodos de selección de enlaces. **A.** Mediantes anillos de cobertura. **B.** K nodos más cercanos.

mayor será la distancia de ese enlace, asignando por tanto un coste mayor en la matriz a los enlaces pertenecientes a los anillos más alejados.

Una implementación que mejora el mecanismo de los anillos es la siguiente: se añade un paso de procesado, que limita el número de conexiones de larga distancia mediante un parámetro configurable (por parte del diseñador) para indicar el número máximo permitido de estas conexiones.

El otro método (figura 3.5B) sirve para conectarse con los *K nodos* más cercanos, asegurando que un nodo siempre vaya a estar conectado con otros. No contempla restricciones de distancia, pudiendo haber casos en que resulte interesante este sistema de conexión (escenarios pequeños por ejemplo). En la matriz de conexiones se almacena la distancia a los nodos conectados para penalizar aquellos enlaces de mayor distancia. .

Ambos métodos presentan ventajas e inconvenientes. Usando los *anillos de cobertura*: una mala configuración del tamaño de los radios puede provocar multitud de posibles conexiones entre nodos, con el esfuerzo que eso acarrearía, o la aparición de nodos aislados que no han encontrado ningún nodo en su zona de cobertura.

El otro sistema (ver figura 3.5B) garantiza que un nodo no quede aislado, pero puede darse el caso de que se formen islas de *K+1* nodos. Para reducir la probabilidad de que aparezcan estas islas se podrá ir aumentando *K*. De este modo para tener una probabilidad del cien por cien que evite la formación de islas, hay que seleccionar un *K* mayor o

igual que la mitad de los nodos de la simulación (`N/2`). Lo cual no tiene mucho sentido debido a que se fuerza al sistema a que cada nodo tenga una cantidad de enlaces muy elevada.

Una característica que diferencia a ambos mecanismos es el tipo de enlaces generados. Con el sistema de anillos los enlaces serán bidireccionales en todos los casos ya que al estar basados en restricciones de distancia y siendo todas ellas iguales para todos los nodos, un nodo que cumpla la condición de alguno de los anillos, será válido a su vez para el otro, bajo la misma condición. En el otro mecanismo, los enlaces son direccionales puesto que un enlace puede tener sus `K` nodos más cercanos y no formar parte de los más cercanos de alguno de ellos. Por lo tanto, en el primer caso nos queda una matriz de adyacencia simétrica y en el otro no.

Otra posibilidad a la hora de establecer conexiones con los vecinos, es por ejemplo, mediante la asignación de preferencias específicas a través de determinados nodos. Siendo útil cuando en la red existen nodos con una mayor capacidad de procesamiento, otorgándoles un menor coste a los enlaces con ese nodo. Atendiendo a los requisitos, prestaciones y necesidades de la red, habrá configuraciones mejores que otras, siendo tarea del diseñador decidir qué método es el que mejor se ajusta al modelo.

El siguiente paso dentro de este bloque, una vez establecidos los enlaces, es el descubrimiento de rutas. Conocidos los enlaces entre los nodos, se aplica un algoritmo de encaminamiento para encontrar las rutas entre los nodos que deseen comunicarse. La elección de los instantes de tiempo en los que se ejecuta el algoritmo corre a cargo del diseñador. Es importante hacer una buena elección de este parámetro, ya que dependiendo del fin de la simulación interesa escoger instantes largos (búsqueda de mejoras en rendimiento y consumo energético) o cortos (estudio de algoritmos).

La búsqueda de vecinos se realiza bajo la condición de que exista algún cambio respecto al anterior instante (se comprueba los cambios en la matriz de adyacencia), estos se producen a consecuencia del gestor de movimientos, detallado en el capítulo 2.

Una vez finalizada la búsqueda de vecinos, el encaminamiento se realiza mediante una implementación del *K-Shortest Path*.

A continuación se detallan los parámetros de entrada y salida de la función:

- Parámetros de entrada:

  - Matriz de costes.
  - Nodo origen.
  - Nodo destino.
  - Número de caminos máximo a buscar.

- Parámetros de salida:
  - Los *K* caminos más cortos.
  - El coste de esos caminos.

Al final, el resultado que arroja este bloque consiste en una matriz con los `K` caminos (siempre que sea posible) de menor coste entre los nodos que tengan una demanda de tráfico y devuelve la lista de los nodos por los que pasa el camino establecido. Por tanto la integración de nuevos algoritmos de encaminamiento resulta sencilla siempre y cuando la salida de éstos sea como la que ofrece la función descrita.

3.4. Evaluación y Conclusiones

Como final a este capítulo, se presenta la salida que ofrece la implementación de este bloque, detallando el significado y la utilidad que tiene.

La importancia de este bloque (el encaminamiento en redes ad hoc) se debe, en gran medida, a que la elección de un mal protocolo, uno que no se ajuste a las necesidades del problema, puede derivar en fallos graves a la hora de ir creando las rutas y por lo tanto impedir que se pueda establecer la comunicación entre nodos. Es necesario contar con una herramienta que permita analizar las prestaciones que ofrecen los distintos algoritmos y aquellos que puedan ser diseñados, en este tipo de redes.

La gestión del establecimiento de los enlaces se irá monitorizado en base a gráficas que describan los enlaces establecidos en cada momento. La gráfica representa un mapa del escenario simulado, por lo que ambos ejes vienen dados en metros y representa a los nodos en la posición geográfica en la que se encuentran y los enlaces que han establecido. Para el mecanismo de *anillos de cobertura* que establece enlaces de diferentes categorías (en función de los anillos implementados), se representan dichos enlaces mediante colores distintos para distinguir unos de otros.

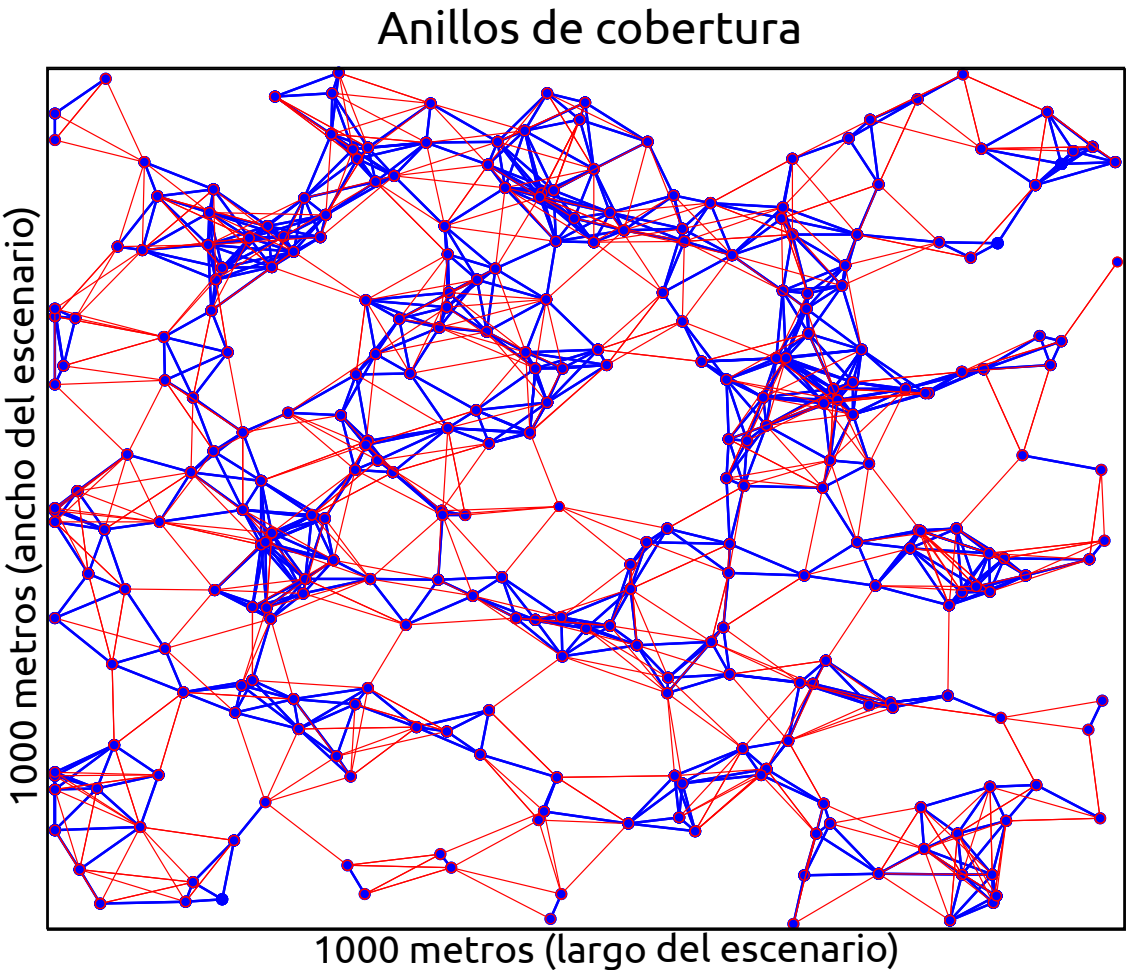
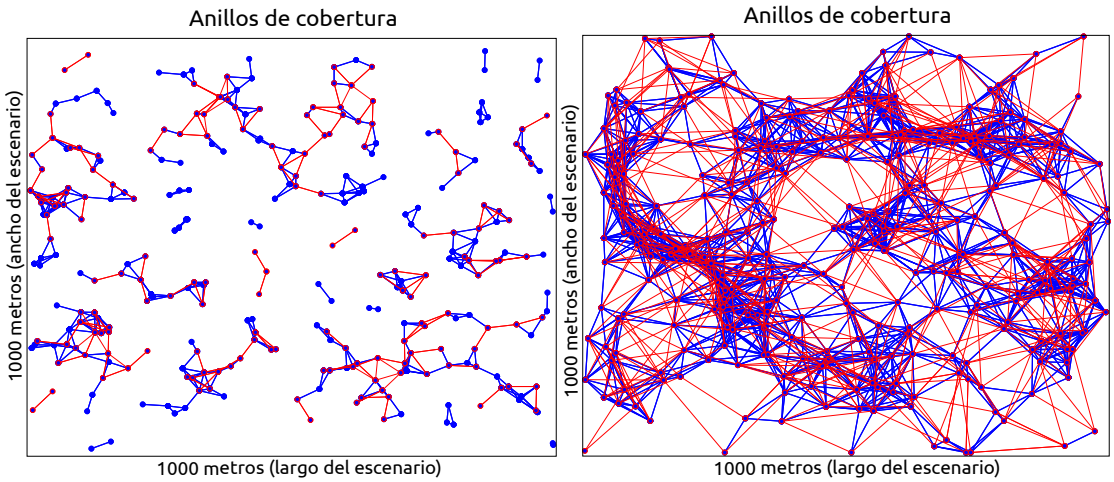


Figura 3.6: Enlaces durante un instante de tiempo mediante el método de selección enlaces de *anillos de cobertura* en una simulación con 300 nodos.

De esta forma se realiza una simulación a modo de ejemplo en la que los nodos se mueven por todo el escenario, de tal forma que en cada intervalo de tiempo realizan una búsqueda de nodos con los que establecer la conexión mediante alguno de los mecanismos explicados. En concreto para el mecanismo de *anillos de cobertura* se han utilizado dos anillos, por lo que ha sido necesario definir tres radios, es decir, el anillo de grado uno comprende entre $(0, r_1]$ y el de grado dos entre $(r_2, r_3]$, siendo $0 < r_1 < r_3$ y r_3 el radio de cobertura total del nodo.

En la figura 3.6 se observa el estado de las conexiones en un instante de tiempo cualquiera, configuradas mediante el mecanismo de búsqueda de vecinos de *anillos de*



(a) Radios asignados (en metros): $0 < 20 < 30$. (b) Radios asignados (en metros): $0 < 60 < 90$.

Figura 3.7: casos particulares del establecimiento de nodos mediante el mecanismo de selección de *anillos de cobertura* en simulaciones con 300 nodos en un escenario de $1000m^2$.

cobertura, se observan dos tipos de enlaces: los de grado uno (en color azul), aquellos que están a una distancia corta; y los de grado dos (en color rojo), nodos pertenecientes al siguiente anillo (más alejado). Como ya se comentó anteriormente, el número de enlaces lejanos está limitado para no saturar de conexiones a los enlaces. Este método utiliza una matriz que tendrá *infinito* para las parejas de nodos que no estén conectados y dos valores (costes asignados a cada anillo), respectivamente en función del anillo al que pertenezca la conexión.

Como ya se ha explicado, este mecanismo requiere de una elección adecuada de los radios de cobertura ya que de lo contrario puede dar lugar a comportamientos no deseados. La figura 3.7 muestra los dos casos opuestos de los que se habla, en la figura 3.7a, los radios seleccionados son pequeños. Aquí entra de nuevo el papel del diseñador para valorar el tamaño de los radios que desea utilizar, por el contrario en la figura 3.7b de la derecha aparece una red saturada de enlaces debido a una elección alta de los radios de cobertura.

El otro tipo de gráfica que ofrece el simulador (ver figura 3.8) corresponde a una simulación en otro instante de tiempo cualquiera con el mecanismo de selección de *K nodos*. Se trata de una red menos densa respecto al número de conexiones, que la ofrecida en la simulación con el sistema de anillos (ver figura 3.6), debido a la elección de un valor bajo del número de vecinos. La matriz generada mediante este método tendrá *infinito* donde

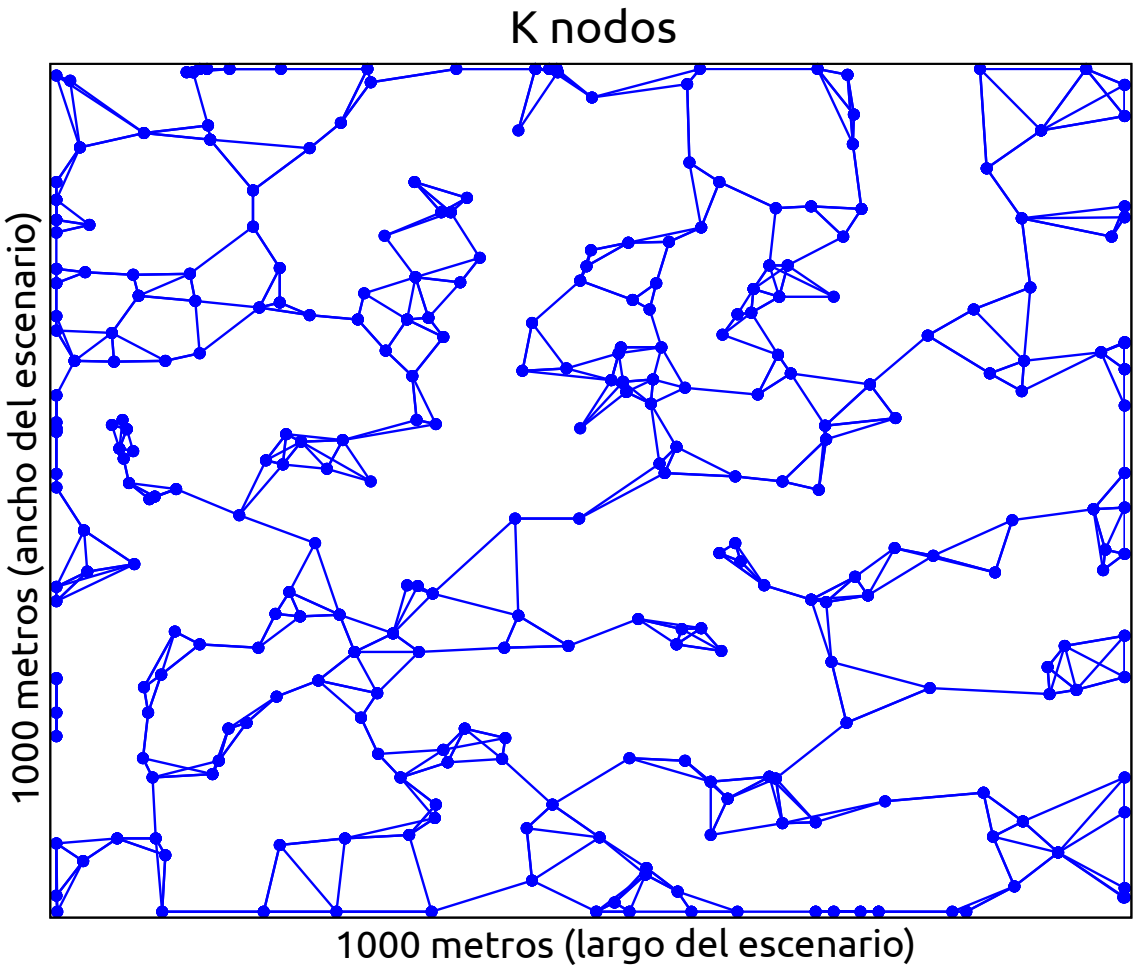
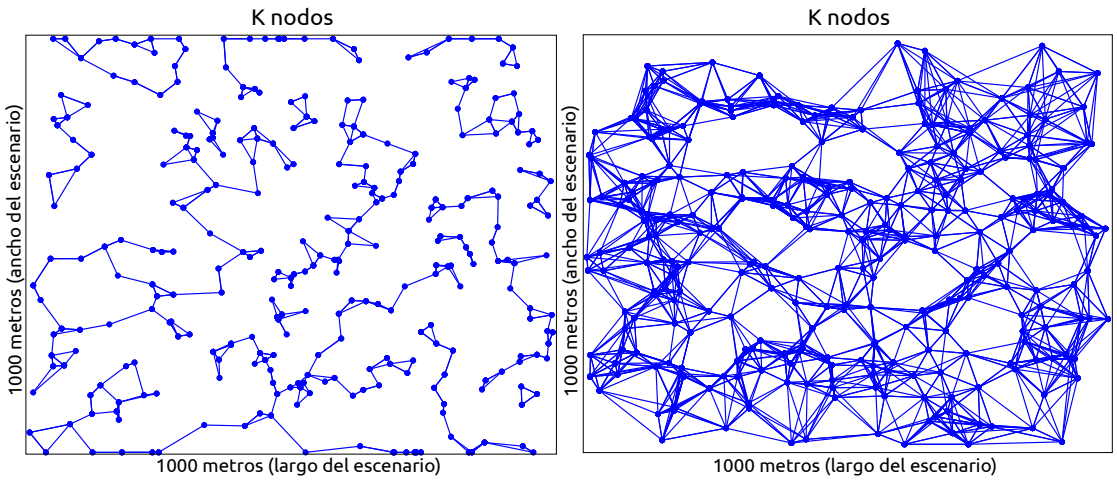


Figura 3.8: Enlaces dados mediante el método de selección enlaces de *K nodos* en una simulación con 300 nodos.

no haya conexión y la distancia del enlace en aquellas posiciones de la matriz donde si la haya.

Al igual que ocurría con el mecanismo de *anillos de cobertura*, el diseñador ha de elegir un valor del número de vecinos adecuado para que la red se comporte de manera deseada. La elección de un valor de demasiado bajo o alto desemboca en problemas de islas de nodos o exceso de enlaces por nodo respectivamente (ver figura 3.9)

La representación gráfica es útil para saber qué está pasando en cada momento, sin embargo, lo necesario para el funcionamiento del bloque y por ende el simulador completo, es la matriz con los datos de los enlaces, la cual una vez se hayan establecido dichas



(a) Número de vecinos: 2.

(b) Número de vecinos: 7.

Figura 3.9: Comparativa método de selección enlaces de *K nodos* en una simulación con 300 nodos en un escenario de $1000m^2$.

conexiones se remplazarán los ceros que tenga por *infinitos*, para representar el coste del enlace en la matriz. Son esos valores los que se utilizan para crear las rutas de la red en la simulación, sobre las cuales el siguiente bloque realiza el proceso de optimización.

Como cierre al capítulo se presenta otro análisis, cuyos resultados se exponen mediante unas gráficas (ver figuras 3.10 y 3.11) correspondientes a un conjunto de simulaciones detalladas a continuación:

- **1.** Simulaciones en un escenario cerrado de 1000 metros de ancho y largo. Analizando el comportamiento de 300 nodos, utilizando el mecanismo *anillos de cobertura* y radios $0 < 70 < 100$ metros.
- **2.** Simulaciones en un escenario cerrado de 1000 metros de ancho y largo. Analizando el comportamiento de 300 nodos, utilizando el mecanismo *K nodos* con $K = 3$.
- **3.** Simulaciones en un escenario comercial de 1000 metros de ancho y largo. Analizando el comportamiento de 300 nodos, utilizando el mecanismo *anillos de cobertura* y radios $0 < 70 < 100$ metros.
- **4.** Simulaciones en un escenario comercial de 1000 metros de ancho y largo. Analizando el comportamiento de 300 nodos, utilizando el mecanismo *K nodos con*

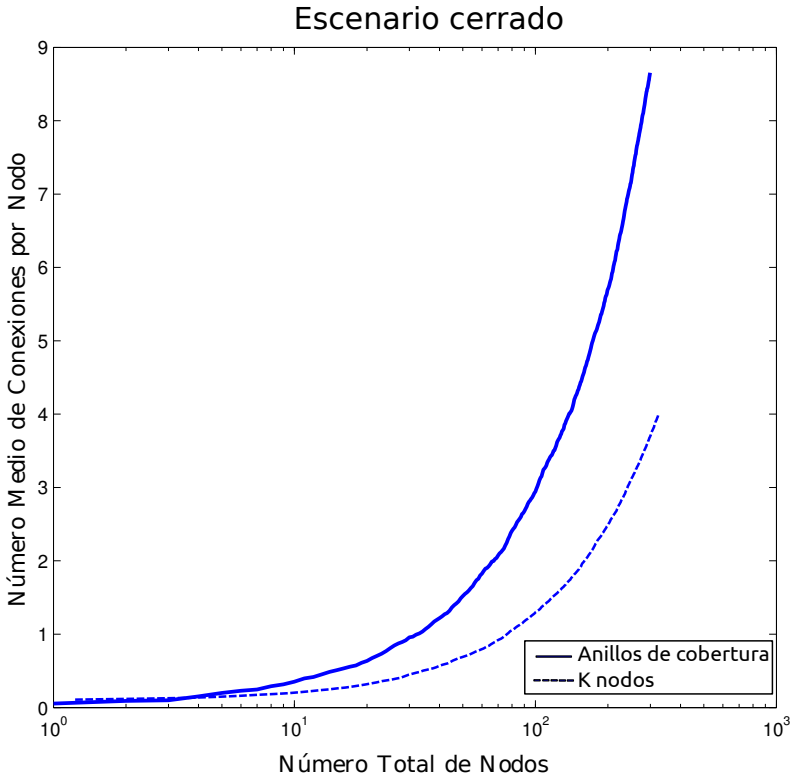


Figura 3.10: Número medio de conexiones en un escenario cerrado $(1000m^2)$. **1.**Recinto cerrado, sistema de *anillos* con radios $0 < 70 < 100$ metros. **2.** Recinto cerrado, sistema de *K nodos* con K=3.

$$K = 3.$$

Cada gráfica presenta el número medio de conexiones establecidas por cada nodo. Por lo tanto el eje X presenta el número total de nodos utilizados para cada simulación y el eje Y indica el número medio de conexiones por nodo.

En la primera gráfica (ver figura 3.10) se representa de forma solapada el número de conexiones medio por nodo utilizando ambos mecanismos en un escenario cerrado, se aprecia un aumento lineal a medida que se incrementa el número de nodos en la simulación.

En la otra gráfica (ver figura 3.11) se expone el número medio de conexiones para ambos mecanismos de selección de enlaces en escenarios comerciales. Se observa un comportamiento similar al del otro escenario de prueba (recinto cerrado).

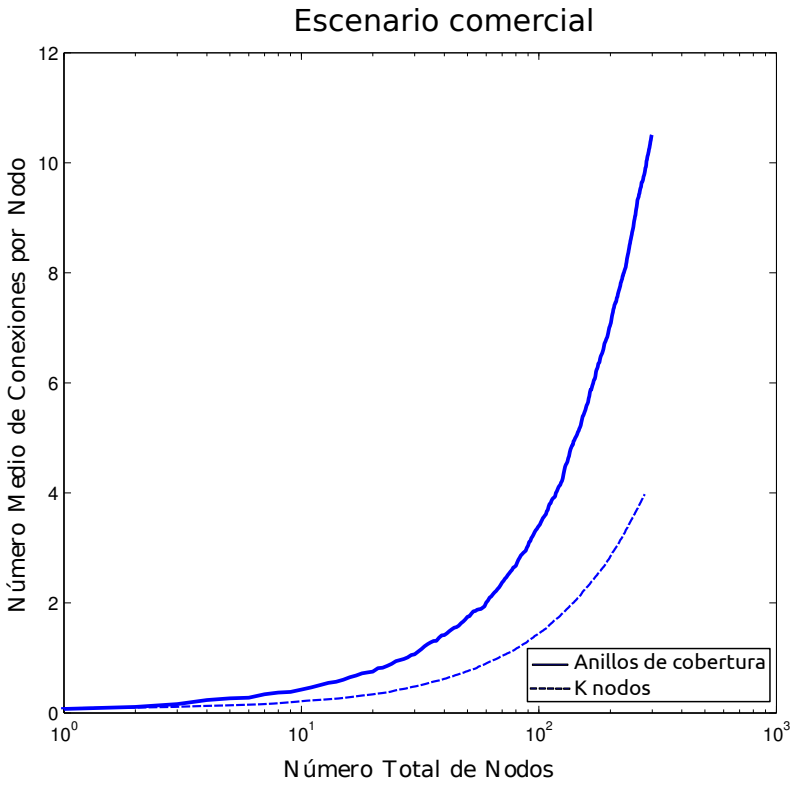


Figura 3.11: Número medio de conexiones en un escenario comercial ($1000m^2$). De izquierda a derecha: **1.** Recinto comercial, sistema de *anillos*. **2.** Recinto comercial, sistema de *K nodos*.

En conclusión, el mecanismo de *anillos de cobertura* establece un número de conexiones mayor, en comparación con el *K nodos*, también se aprecia cómo la configuración del escenario de tipo comercial, hace que los nodos tengan un mayor número de enlaces (con el mecanismo de *anillos de cobertura*), ya que al estar concentrados con mayor probabilidad por las zonas de tiendas hace que los nodos se encuentren menos dispersos y por tanto se alcancen unos a otros con mayor facilidad. Con estas gráficas el diseñador puede estimar los radios de cobertura o número de enlaces deseado, para el estudio o simulación que desee realizar.

Capítulo 4

Optimización Convexa

En este capítulo se explica el tercer bloque de este proyecto concerniente a la optimización del tráfico. Se realiza una introducción a la optimización convexa y se presenta la herramienta utilizada

4.1. Introducción

La optimización [16] es una herramienta esencial en la toma de decisiones basada en criterios cuantitativos. El concepto de convexidad, que se aplica tanto a conjuntos como a funciones, es fundamental en optimización, pues implica que el problema formulado es accesible en diferentes aspectos. Además, la clasificación de un problema como perteneciente a la clase de los problemas de optimización convexa presenta dos grandes ventajas: que dichos problemas se resuelven numéricamente de manera eficiente y que se alcanzan soluciones globales del problema.

Por tanto el objetivo final de este proyecto es aportar una herramienta que permita aplicar técnicas de optimizado para estudios de costes sobre redes ad hoc. De forma que el estudio de los resultados arroje posibles soluciones que mejoren la congestión que pueda surgir en un momento dado.

4.2. Estudio del Estado del Arte

4.2.1. Fundamentos de la Optimización

El concepto de optimización puede ser aplicado a la mayoría de las disciplinas. Cualitativamente, esta afirmación implica la necesidad de tener múltiples opciones de decisión; reconociendo implícitamente la necesidad de elegir entre diferentes alternativas. En este proyecto se ha tratado la optimización de una forma cuantitativa, lo que significa que los resultados de aplicarla al problema o diseño arrojarán números que definirán la solución, en otras palabras, números o valores que caracterizarán ese diseño en particular.

Una descripción cuantitativa de la solución requiere una descripción cuantitativa del problema en cuestión. Esta descripción es lo que se conoce como el modelo matemático. El diseño, caracterización y circunstancias deben ser expresadas matemáticamente. Cada problema necesita información específica del área al que se refiere, esto hace necesario que el diseñador tenga unos conocimientos previos en dicho campo.

Este modelo matemático requiere de una serie de elementos para su formulación y posterior procesamiento [17]:

- **Variables de diseño**: son entidades que identifican un diseño en particular. En la búsqueda del diseño óptimo estas entidades irán cambiando dentro de un rango preestablecido. Los valores de un conjunto completo de estas variables caracteriza un diseño específico. El número y tipo de entidades pertenecientes a este conjunto son de suma importancia en la identificación y creación del problema de diseño cuantitativo. Es básico que la elección capture la esencia del objeto que esta siendo diseñado y al mismo tiempo provea una caracterización cuantitativa de éste. Estas variables vienen a ser las incógnitas del problema a resolver y la elección de las mismas correrá a cargo del diseñador. Existe un requisito fundamental que debe cumplir este conjunto de variables y es que deben ser linealmente independientes, lo que viene a significar que no se puede determinar el valor de una de las variables de diseño a partir del resto mediante operaciones aritméticas básicas (escalado o suma) y por tanto esta propiedad garantiza que el conjunto de variables escogido es el mínimo posible. Esto es algo importante ya que el esfuerzo de obtener la solución varía como una potencia entera del número de variables.

- **Parámetros de diseño**: identifican las constantes que no cambiarán para ninguno de los diseños que sean comparados. Son por tanto parámetros de poca importancia ya que no juegan ningún papel en la mayoría de diseños óptimos.

- **Funciones de diseño** (una o varias): definen la información del diseño más relevante. Están evaluadas usando las variables y parámetros de diseño explicados anteriormente, estableciendo el modelo matemático del problema. Pueden representar objetivos (uno o varios) y/o restricciones. El cumplimiento de estas garantiza la validez del diseño. Si no se indican explícitamente, el diseñador es responsable de la identificación de estas.

- **Funciones objetivo** (una o varias): el diseño de un problema de optimización típico queda definido usando una única función objetivo. Normalmente el formato de esta declaración es para minimizar o maximizar alguna cantidad que se calcula usando alguna función de diseño. La función objetivo va a depender, explícita o implícitamente, de las variables de diseño.

- **Funciones de restricción** (una o varias): como en las funciones de diseño, estas también estarán influenciadas por las variables de diseño. El formato de estas funciones requieren que sean comparadas con algún limite numérico. Dicho valor permanecerá constante a lo largo de la optimización. La comparación será dada por algún operador relacional. Las funciones de restricción se pueden clasificar como restricciones de igualdad o restricciones de desigualdad. Hacer esta distinción es debido a que a la hora de buscar la solución óptima se tratan de distinta manera.

    - Restricciones de igualdad: Matemáticamente las restricciones de igualdad son más fáciles de manejar. Sin embargo, numéricamente requieren un mayor esfuerzo para que se satisfagan, puesto que son más restrictivas. Puede haber más de una restricción de este tipo. El número de variables de diseño debe ser mayor que el de restricciones de igualdad para que la optimización tenga lugar. Si el número es igual, entonces el problema será resuelto sin hacer referencia al objetivo. En caso de que sea menor, podría dar lugar a una definición inconsistente del problema. Estas restricciones han de ser linealmente independientes.
    - Restricciones de desigualdad: Es más común que aparezcan estos tipos de restricciones en la formulación de problemas. Ofrecen una mayor flexibilidad en la selección del diseño.

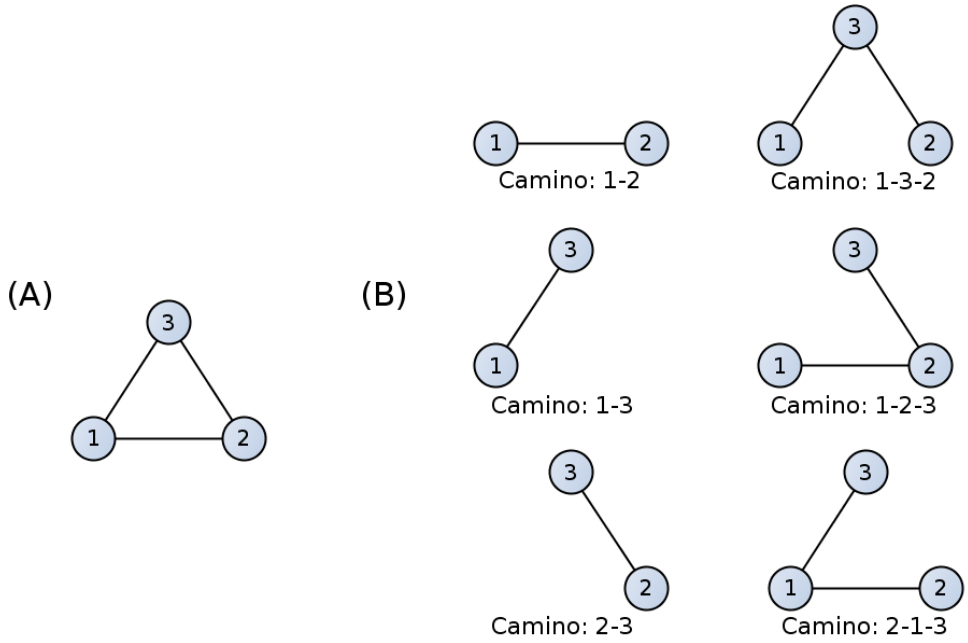


Figura 4.1: (A) Ejemplo de una red con tres nodos y (B) Todos los posibles caminos para los nodos de este ejemplo

- **Restricciones de borde**: expresan los límites para las variables de diseño. Cada variable de diseño debe estar contenida por valores numéricos tanto en el límite superior, como en el inferior. Se debe hacer la elección de estos límites en base a lo que se consideraría un diseño aceptable.

4.2.2. Diseño de Problemas de Red

Los problemas de diseño de red pueden ser formalmente formulados usando notación matemática. Es necesario escoger una buena notación que nos permita representar un problema de diseño de una forma compacta y sin ambigüedades. Y además, que ayude al entendimiento del problema en cuestión. Para seguir este capítulo se detallará la notación escogida mediante un ejemplo sencillo [18] (ver figura 4.1).

En la figura 4.1 (A) se ve una red formada por tres nodos. El concepto de volumen de demanda hace referencia al volumen de tráfico o el ancho de banda requerido entre un par de nodos, dependiendo del tipo de red sobre la que se este trabajando. La demanda

entre el nodo 1 y 2 es de cinco unidades. Entre 1 y 3 es de siete unidades y la que hay entre 2 y 3 es de ocho unidades, por lo que existirán tres demandas, $D = 3$ y $d = 1,2,3$. En este caso se tratan de demandas totales entre nodos, y no demanda de un nodo a otro. Se representa el volumen de demanda con h_d. De esta forma quedan las siguientes demandas:

$$h_1 = 5, \;\; h_2 = 7, \;\; h_3 = 8.$$

Se puede ver fácilmente que los subíndices se usan para identificar el índice de la demanda, existen otras notaciones en la que los subíndices de las demandas indican los nodos finales de la misma, pero pueden llegar a confundir. En base a esta información y a la figura 4.1 (B) se deduce que cada una de las demandas puede ir por dos caminos diferentes, $P_d = 2$, $p = 1,2$. En el caso concreto de la demanda 1, $P_1 = 2$ ya que existen dos rutas, $1-2$ y $1-3-2$ posibles para encaminar. Cuanta demanda se va a mandar por cada camino depende del diseño de red objetivo (que se tratará más adelante). Para denotar estas incógnitas, que representan las variables de flujos de demanda por camino, usaremos x_{dp}, indicando que es la cantidad de demanda *d* que se encamina por la ruta *p*.

$$x_{11} + x_{12} = 5 \quad (= h_1)$$
$$x_{21} + x_{22} = 7 \quad (= h_2)$$
$$x_{31} + x_{33} = 8 \quad (= h_3)$$

De forma análoga se usa E para denotar el número de enlaces en la red y e para etiquetarlos:

$$enlace \; 1-2 \longleftrightarrow e = 1.$$
$$enlace \; 1-3 \longleftrightarrow e = 2.$$
$$enlace \; 2-3 \longleftrightarrow e = 3.$$

Por otro lado la capacidad de cada enlace queda representada de la siguiente manera (se asumen los siguientes valores para el ejemplo):

$$c_1 = 10, \;\; c_2 = 10, \;\; c_3 = 15.$$

Para definir el diseño, queda por último tomar una función objetivo. Puede hacer referencia a cualquiera de las distintas partes del diseño, son funciones para minimizar o maximizar, algunos ejemplos son:

- Minimizar el coste total de encaminamiento.
- Minimizar el tráfico en el nodo más congestionado de la red.
- Maximizar el tráfico por los enlaces que nos interese (bajo nuestro control, etc).
- Minimizar la capacidad de los enlaces. Para reducir costes de instalación.

Para el ejemplo explicativo se utiliza una función coste que minimiza el coste total de encaminamiento (asumiendo que el coste de salto de cada enlace es uno). Por lo que se obtiene la siguiente función de costes:

$$F = x_{11} + 2x_{12} + x_{21} + 2x_{22} + x_{31} + 2x_{32}$$

Las funciones de coste, pueden tener una complejidad elevada, en este caso no es así para facilitar la comprensión de las partes involucradas. Sin embargo, puede que se den situaciones en las que interese manejar más variables, por ejemplo en caso de querer minimizar capacidades de los enlaces habría que incluir otro conjunto de incógnitas (representadas con otra letra, p.e. y) en la función objetivo. Apareciendo también en las condiciones planteadas.

Por tanto, con la notación escogida se puede plantear este problema de diseño de forma compacta de la siguiente manera:

minimizar:

$F = x_{11} + 2x_{12} + x_{21} + 2x_{22} + x_{31} + 2x_{32}$

condicionado a:

$x_{11} + x_{12} = 5$
$x_{21} + x_{22} = 7 \;\; (demandas)$
$x_{31} + x_{33} = 8$

$$x_{11} + x_{22} + x_{32} \leqslant 10$$
$$x_{12} + x_{21} + x_{32} \leqslant 10 \quad (capacidades)$$
$$x_{12} + x_{22} + x_{31} \leqslant 15$$

$$x_{11}, x_{12}, x_{21}, x_{22}, x_{31}, x_{32} \geq 0 \ (restricciones\ de\ borde)$$

4.3. Descripción del Bloque de Optimización Implementado

4.3.1. Estructura del Bloque

El bloque de optimización a nivel teórico es el más denso de los tres y propone una gran funcionalidad. Hace que el usuario sin preocuparse de la implementación de la red o de tener que configurar las rutas, sea capaz de probar diferentes funciones de coste para poder optimizar el tráfico de la red: controlar el volumen de las demandas existente entre los nodos, minimizar tráfico entre enlaces durante la simulación, etc. Hay que ser consciente de que para crear la función de coste es necesario un conocimiento de la red que se va a simular puesto que el número de variables varía en función de los nodos y enlaces en la red.

Lo que se pretende es ofrecer un mecanismo que permita obtener los valores mínimos de una función a partir de un problema dado. Viendo la figura 4.2 se observa que las entradas de la izquierda serán en parte aquellas que se han tenido que usar en el anterior bloque (capítulo 3) y por lo tanto ya estarán definidas y resultados obtenidos del mismo. En la parte superior de la misma tenemos el nuevo parámetro, la función objetivo. La salida de este bloque debería ser por tanto los valores que hacen mínimo el problema planteado.

Detalles específicos de adecuación de datos de entrada:

La entrada de datos provenientes del bloque previo (bloque de encaminamiento), obliga a realizar unos pasos previos para poder hacer uso de la herramienta de optimizado que se explicará más adelante. Los parámetros de entrada diseñados para llevar a cabo

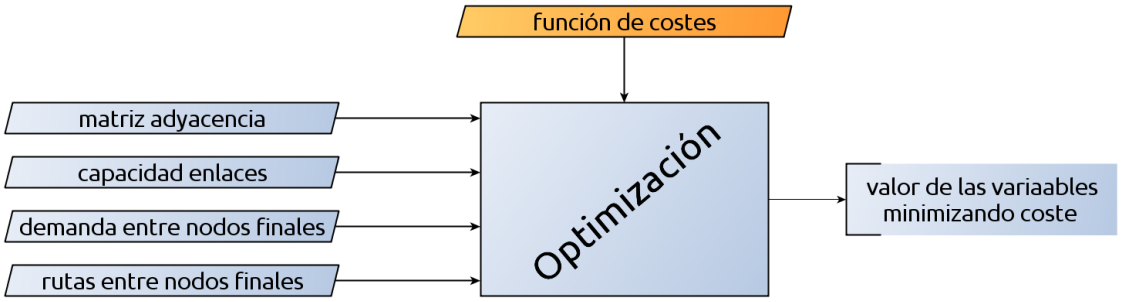


Figura 4.2: Esquema general del bloque de optimización

este bloque son los siguientes:

- **Matriz de adyacencia:**

  La implementación del simulador, como se ha visto en el capítulo de encaminamiento, hace que la matriz de adyacencia se aleje de su definición original, asemejándose más a una matriz de costes entre enlaces conectados.

- **Mecanismo de adaptación de las funciones de restricción:**

  Para iniciar el proceso de optimizado es necesario establecer el modelo matemático debidamente. Para ello esta parte del simulador ha de ser capaz de generar las funciones de restricción tanto de igualdad como desigualdad, a partir de los parámetros de entrada (figura 4.2).

  Referente a la función de restricción de igualdad, se crea una matriz de dimensión `[D×flujos]` a partir de los datos obtenidos en el bloque de encaminamiento (demandas y rutas entre nodos finales), donde `D` es el número de enlaces de la red y `flujos` hace referencia a la suma de todos los flujos de demanda por camino (x_{dp}) existentes. De forma que se representa en la matriz, mediante ceros y unos, los flujos pertenecientes a cada demanda. Colocando un `1` en aquellas columnas donde el flujo corresponda a la demanda de la fila.

  La función de restricción de desigualdad representa los flujos de demanda que pasan por los enlaces de la red. Por ello a partir de la matriz de caminos entre nodos con demanda (rutas entre nodos finales) y la capacidad de los enlaces, se construye una matriz de dimensión `[E×flujos]` en la que `E` se refiere al número de enlaces de la red y `flujos` a la suma de todos los flujos de demanda por camino

existentes. De forma que la matriz tiene `1` en las posiciones donde el flujo (columna) pase por el enlace correspondiente (fila). El resto de posiciones tendrán valor cero.

- **Mecanismo de adaptación de las variables de diseño:**

  Es necesario presentar las capacidades de los enlaces y las demandas entre nodos de manera adecuada para llevar a cabo la optimización de la red. Se trata de las capacidades de los enlaces, asociadas a las restricciones de desigualdad y las demandas entre nodos finales, asociadas a las de igualdad.

Tras este proceso se procede a la optimización propiamente dicha. Para la cual, se hará uso de la herramienta de optimización descrita a continuación.

4.3.2. Herramientas Utilizadas

Para realizar la parte concerniente a la optimización [19], este proyecto hace uso de la función *fmincon*. Integrada dentro del entorno de desarrollo MATLAB [20], forma parte de la caja de herramientas (*toolbox*) de optimización. Se trata de una función muy compleja que busca el mínimo de una función no lineal de varias variables sujeta a una serie de restricciones, es decir, minimiza un problema dado por:

$$\min_x f(x) \text{ dado } \begin{cases} c(x) \leq 0 \\ c_{eq} = 0 \\ A \cdot x \leq b \\ A_{eq} \cdot x = b_{eq} \\ lb \leq x \leq ub \end{cases}$$

Donde $c(x)$ y c_{eq} son las restricciones de desigualdad no lineales y las restricciones de igualdad no lineales respectivamente. A y b son las condiciones lineales de desigualdad, y, A_{eq} y b_{eq} hacen referencia a las condiciones lineales de igualdad. Por último, los límites en los que ha de estar contenida la solución vienen dado por lb y ub.

Al tratarse de una función con estas características, la sintaxis y el número de parámetros es elevado. No tiene sentido para este proyecto describir exhaustivamente todos y cada uno de ellos ya que pueden encontrarse en [21]. Se explicarán, a continuación, aquellos parámetros que se han utilizado para este estudio.

La llamada a la función es: `[x, fval, exitflag] = fmincon(fun, x0, A, b, Aeq, beq, lb, ub, nonlcon, options)`

- `fun`: es la función a ser minimizada o función objetivo $f(x)$. `fun` puede ser un objeto que se le pase a la función directamente o bien un fichero de ejecución (con extensión `.m`) que realice las operaciones necesarias para computar $f(x)$.
- `x0`: punto inicial, es un vector que contiene los valores iniciales asignados a los elementos de x. Son la solución del problema. Representan los flujos de cada demanda, por tanto indicará la cantidad de tráfico a enviar por cada interfaz de salida de los nodos.
- `A` y `b`: esta matriz y vector son, respectivamente, los coeficientes de las restricciones de desigualdad lineal y el vector que incluye los valores de la parte derecha de las desigualdades. En este proyecto, representan los flujos de demanda que pasan por cada enlace (`A`) y las capacidades de estos (`b`). ($A \cdot x \leq b$).
- `Aeq` y `beq`: corresponden a la matriz y al vector que representa las restricciones de igualdad lineal. En este proyecto se utilizan para los flujos de demanda de cada nodo (`Aeq`) y las demandas entre nodos `beq`. ($Aeq \cdot x = beq$).
- `lb` y `ub`: vectores de los límites inferior y superior que se permiten a los valores de los elementos del vector x. Normalmente son del mismo tamaño que x. Cuando no se impongan estos límites, los valores del vector x, pueden dar valores como *–Inf* o *Inf* para cualquier x_{dp}.
- `nonlcon`: función que computa las desigualdades no lineales y las restricciones de igualdad.
- `options`: Agrupación de parámetros que permite configurar opciones del algoritmo.
- `x`: Es la solución encontrada por la función de optimización que cumple las restricciones y minimiza la función.
- `fval`: Devuelve el valor de la función objetivo para la solución `x`.
- `exitflag`: Indica la razón por la que el algoritmo ha terminado, es la condición de salida de la función.

4.4. Evaluación y Conclusiones

La optimización puede estar enfocada a distintos requisitos y prestaciones. Éstas quedan recogidas en diferentes funciones de coste que son las que minimiza la función `fmincon`. En este proyecto se ha trabajado con la función de costes que penaliza el número de saltos para poder realizar pruebas del funcionamiento completo de la herramienta, sin embargo existen otras posibilidades, tales como, funciones para penalizar el balanceo de carga (cuanto más interfaces de salida utilice el nodo mayor consumo), para optimizar el tráfico en función de la distancia de los enlaces (puede que interese dar más saltos con menor coste de transmisión), o para reducir la energía global (reutilizando interfaces o enlaces). La integración de distintas funciones de coste corre a cargo del diseñador, ya que es el que conoce las necesidades reales del sistema a simular. No es objeto de este proyecto la realización de un análisis comparativo de las distintas funciones de coste evaluando cuales son más útiles o convenientes para este tipo de redes.

El bloque ofrece el resultado final de la simulación de forma numérica y gráfica, permitiendo su estudio y evaluación, y dando una idea visual de este.

Para mostrar el funcionamiento y utilidad de este bloque se presenta la optimización llevada a cabo en un instante de tiempo aleatorio para una simulación de una red de cinco nodos (se toma un número pequeño de nodos para facilitar la comprensión del ejemplo). En la figura 4.3 se observa la configuración de la red en ese instante; representa el momento posterior al encaminamiento, por lo tanto los enlaces directos y las rutas se encuentran ya definidos.

La simulación se ha realizado utilizando el mecanismo de selección de enlaces de *anillos de cobertura*, con dos anillos por lo tanto la red presenta enlaces de dos tipos, los de grado uno (azules) y los de grado dos (rojos), que serán los de mayor distancia.

Los enlaces se etiquetan como muestra la figura 4.4. Cada fila corresponde a un enlace e indica el par de nodos que lo forma.

Se asignan unos valores arbitrarios de capacidad de enlaces y demandas entre nodos para el ejemplo de esta parte del proyecto. Son las variables de restricción explicadas anteriormente, y siguiendo la nomenclatura de la función `fmincon`, corresponden a los parámetros `b` y `beq` respectivamente. De forma que cada posición de `b` indica la capacidad de un enlace. Queda por tanto el mapa de enlaces definido de la siguiente manera:

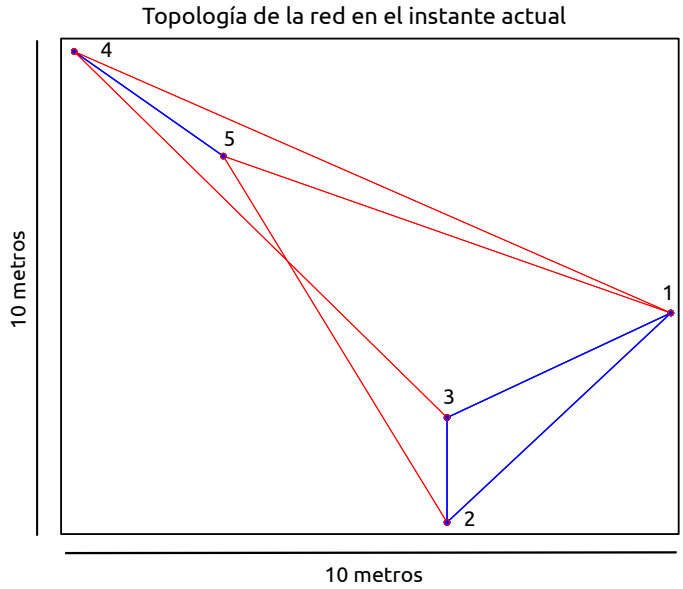


Figura 4.3: Estado de la red finalizado el proceso de encaminamiento.

```
enlaces =

     1     2
     1     3
     1     4
     1     5
     2     3
     2     5
     3     4
     4     5
```

Figura 4.4: Enlaces establecidos.

$$enlace\ 1-2 \rightarrow c_1 = 54.$$

$$enlace\ 1-3 \rightarrow c_2 = 54.$$

$$enlace\ 1-4 \rightarrow c_3 = 6.$$

$$enlace\ 1-5 \rightarrow c_4 = 6.$$

$$enlace\ 2-3 \rightarrow c_5 = 54.$$

$$enlace\ 2-5 \rightarrow c_6 = 6.$$

$$enlace\ 3-4 \rightarrow c_7 = 6.$$

$$enlace\ 4-5 \rightarrow c_9 = 54.$$

Y `beq` corresponde a la demanda de tráfico entre los nodos. En este ejemplo se ha simulado una demanda básica entre todos los nodos de 0.4 unidades y cuatro picos de demanda de 2, 5, 6 y 7 unidades. Cada posición del vector da la demanda de un nodo con otro, se sabe que hay demandas entre todos los nodos, por lo tanto existen $N \cdot (N-1)$ demandas (en el ejemplo $N = 5$). Se ha decidido que el algoritmo de encaminamiento busque un máximo de tres caminos por demanda, por lo que se crean un total de $N \cdot (N-1) \cdot 3 = 60$ caminos.

Tanto las matrices que representan las funciones de restricción `A` y `Aeq` como los vectores `b` y `beq`, debido a su tamaño, se exponen y explican en detalle en el apéndice B.

Por último, se realiza el proceso de optimización que calcula los flujos de demanda para cada una de las demandas entre nodos (existen tantos flujos como caminos haya entre los nodos implicados, en el ejemplo se han establecido tres caminos por demanda). La función, en caso de que exista, devolverá los valores que cumplen y minimizan el problema propuesto (en el apéndice B se muestra el ejemplo de la salida numérica).

De forma visual se ha implementado una gráfica que muestra la carga de los enlaces (figura 4.5). Representado por el siguiente esquema de colores:

- Verde: enlaces con una carga menor al 25 %.
- Amarillo (a rayas): enlaces con una carga entre el 25 % y el 75 %.
- Rojo (punteado): enlaces con una carga superior al 75 %.

Tras este proceso el bloque de optimización concluye. Este proceso se repite a intervalos de tiempo al igual que en la parte de encaminamiento. Para el desarrollo de este proyecto se ejecuta en instantes de tiempo específicos. Puede resultar útil ver la evolución de los flujos de demanda, junto con los cambios de la red para analizar las prestaciones y comportamientos de la misma.

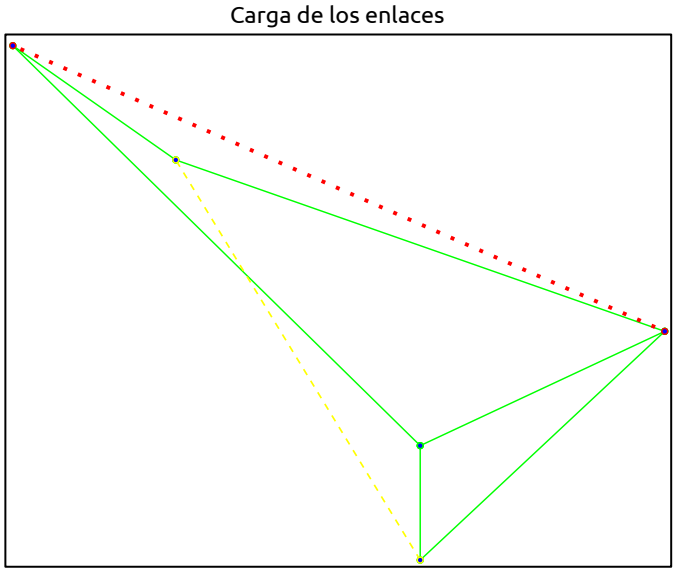


Figura 4.5: Esquema general del bloque de optimización.

Capítulo 5

Integración Sistema Completo

Este capítulo muestra la integración de las tres partes principales del simulador. Ofrece una visión del sistema completo a través de un ejemplo basado en datos reales.

5.1. Simulador Completo

El esquema 5.1 muestra la integración de las partes fundamentales de las que consta este simulador. A la vista de los parámetros de entrada presentes en él, es fácil hacerse una idea de la versatilidad que ofrece.

Definir claramente las entradas y salidas de cada bloque hace que la integración de las tres partes sea cómoda y sencilla. Como se explica en los anteriores capítulos la ejecución de la simulación se basa en eventos. Durante la fase de diseño ha estado presente esta necesidad para permitir realizar un análisis más detallado de los datos cuando sucede un evento crítico para el sistema, en contra partida con un simulador basado en tiempo continuo con el que se obtendrían valores instantáneos, a costa del coste computacional que introduce el procesado continuo de todo el sistema. Sin embargo, dado que lo interesante del estudio ocurre en los momentos de cambio del sistema, merece la pena inclinarse por la opción de un simulador basado en eventos.

Además de los eventos propios del sistema, ciertas partes del simulador entran en funcionamiento a intervalos de tiempo discretos y periódicos. En estos instantes se realizan operaciones de los tres bloques. El bloque de modelado se ejecuta en todas los intervalos de la simulación, para tener constancia de la posición de los nodos en todo momento. En los otros dos bloques (encaminamiento y optimización), correrá a cargo del diseñador la

elección del tiempo de ejecución correspondiente.

5.2. Simulación Escenario Real

Parametrización del escenario:

La figura 5.2[1] ofrece un ejemplo real del escenario objetivo sobre el que se va a modelar la simulación. En este caso se trata de una avenida comercial cuyo modelo tiene las siguientes características (figura 5.3):

- Tamaño: un ancho de 20 metros y una longitud de 200 metros.
- Número de usuarios: 20 nodos.

Las características particulares al movimiento de las personas se detallan a continuación:

- Velocidad: comprendida entre 1 km/h y 3 km/h.
- Tiempo de pausa: comprendida entre 1 minuto y 10 minutos (las zonas comerciales tendrán un tiempo de pausa mayor).
- Distancia de trayecto: Comprendida entre 1 metro y 500 metros.

Respecto a la simulación general se establece:

- Tiempo de simulación: 5 horas.
- Tiempo de intervalo: 5 minutos.

Para esta la simulación se selecciona de entre los posibles el modelo Random Waypoint modificado.

[1] http://www.flickr.com/photos/grumpywolf/3032497075

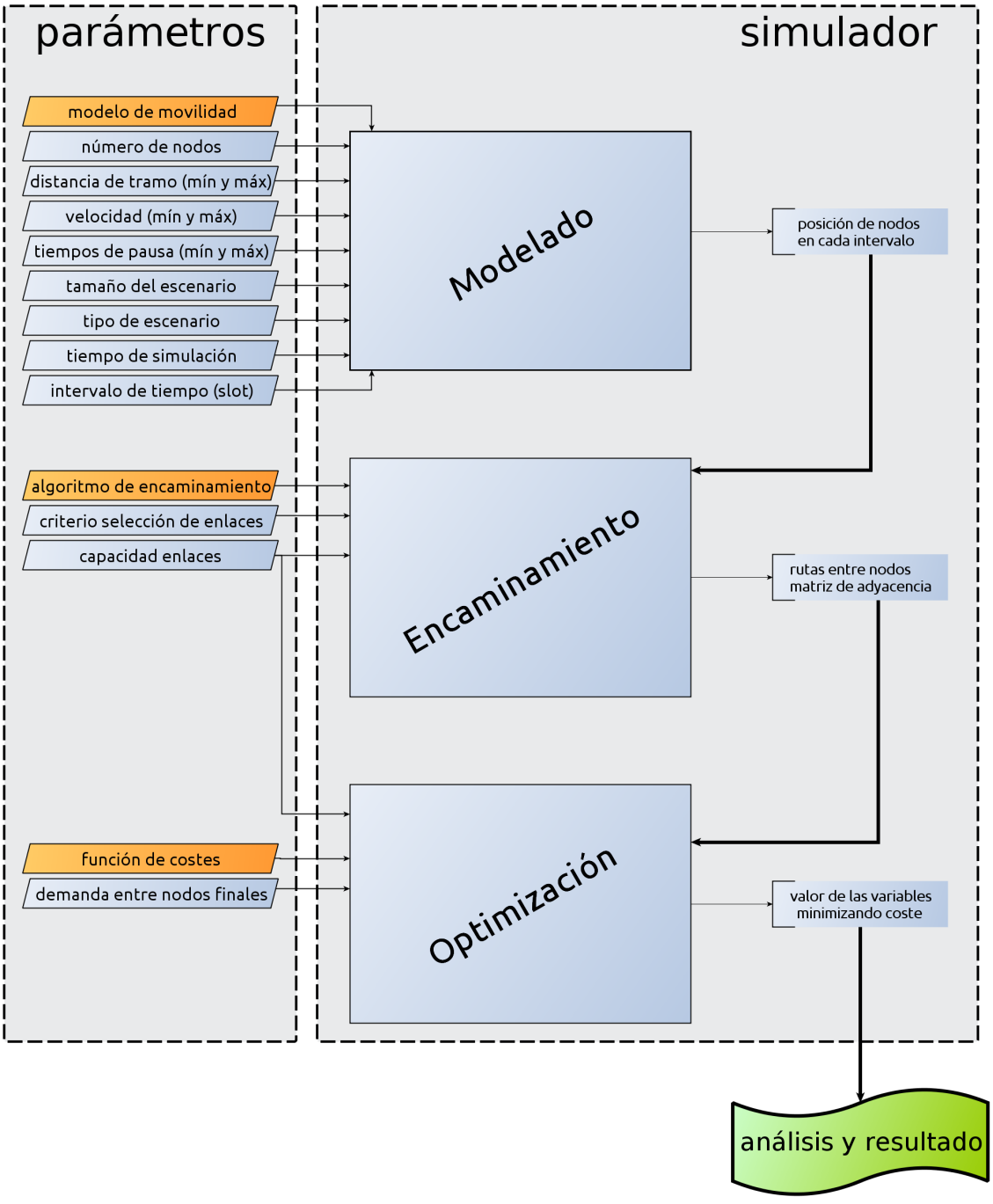


Figura 5.1: Esquema del simulador completo.

Figura 5.2: Ejemplo de avenida comercial. Autor de la foto: Daniel Villoldo.

Figura 5.3: Ejemplo real: Escenario simulado.

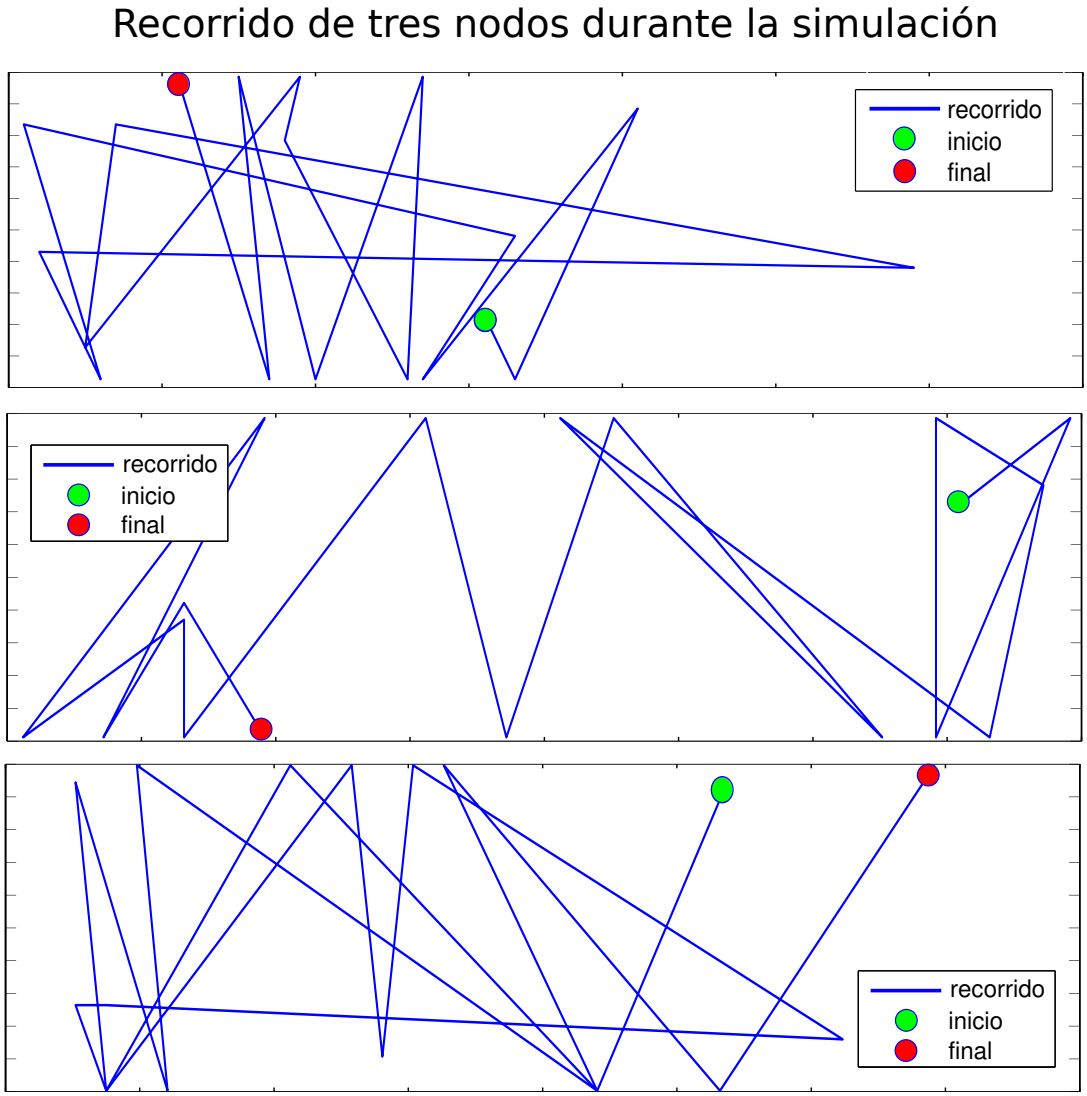


Figura 5.4: Recorrido de tres nodos durante la simulación.

Resultados parciales del modelo de movimiento:

Respecto al análisis del modelado que se realiza en la simulación llevada a cabo, se presenta el recorrido que han descrito tres personas elegidas al azar de las veinte totales. De esta forma se observa el mismo comportamiento en todas ellas, y el grado de aleatoriedad que provee este modelo (ver figura 5.4) y que como ya se ha comentado en el capítulo 2 se ajusta a lo esperado en este escenario.

Durante el transcurso de la simulación, se analiza en instantes de tiempo periódicos concretos la densidad instantánea que existe en el escenario. Al tratarse de una simulación en una zona comercial, aparecerán concentraciones de nodos con mayor probabilidad por las zonas de tiendas (borde superior e inferior, figura 5.3). Se puede observar cómo va cambiando las zonas densas del escenario en los cuatro instantes de tiempo en los que se ha tomado dicha medida (ver figura 5.5).

Por último, se presentan los datos estadísticos obtenidos de la simulación (ver figura 5.6). Se observa que la distancia media que recorren los nodos se estabiliza en torno a los 5 metros y medio, la media obtenida no se corresponde con la media del valor de inicialización de distancia de trayecto (250 metros) ya que esta hace referencia a la distancia que recorrerían los nodos en un espacio abierto, sin tener en cuenta las condiciones del escenario. Al tratarse de un recinto comercial de estas dimensiones la probabilidad de que haya trayectos largos sin parada es muy baja. Además para simular la existencia de zonas de absorción de la atención de los usuarios, se han inicializado los nodos con tiempos de pausa elevados (simulando zona de tiendas). Éstos, al entrar en dichas zonas, pasarán a estado de pausa (que en nuestro caso es del orden de duración de los intervalos), y por eso el valor medio obtenido es menor.

En cuanto al tiempo de pausa medio resultante, se observa cómo tras un transitorio de tiempo de 75 minutos, también se estabiliza. Dicho transitorio puede ser mayor o menor en función de las condiciones de partida con las que se inicie la simulación. El tiempo medio de pausa resultante en nuestro ejemplo es de unos 9 minutos, siendo el tiempo medio introducido para el pasillo comercial de entre 1 y 10 minutos. Al igual que ocurre con la distancia, nuestro escenario condiciona el comportamiento de los nodos elevando la media del tiempo de pausa debido a la alta probabilidad de que ocurra un evento de parada en cada intervalo.

Resultados periódicos del encaminamiento de los nodos:

En el ejemplo se ha seleccionado el mecanismo de selección de enlaces de *anillos de cobertura* para el descubrimiento de vecinos. En los eventos de tiempo seleccionados se establecerán los enlaces con los nodos que estén dentro de las zonas de cobertura de cada uno.

Para la simulación realizada se han configurado dos anillos de cobertura. Con las

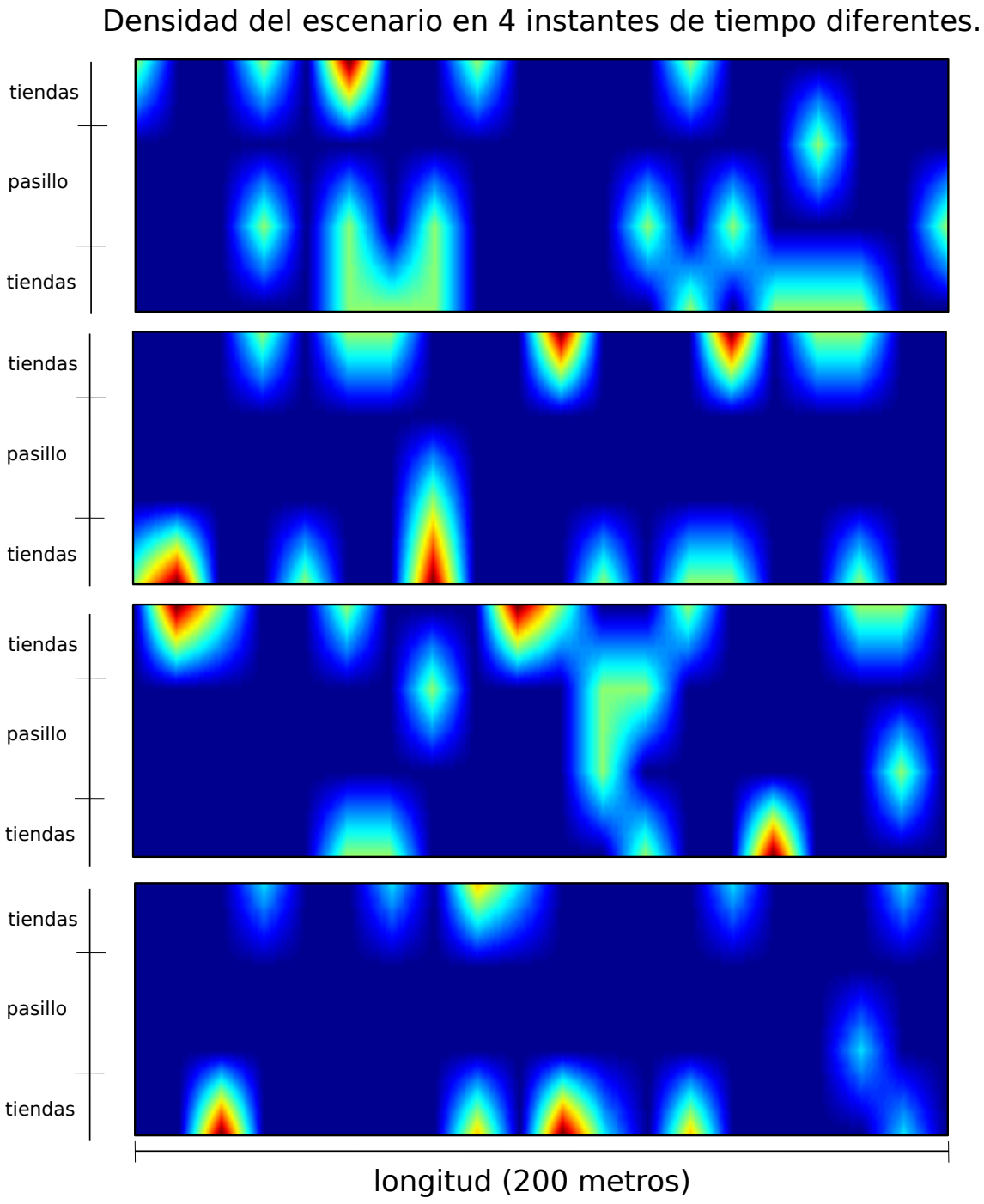


Figura 5.5: Densidad de las personas distribuidas por el pasillo.

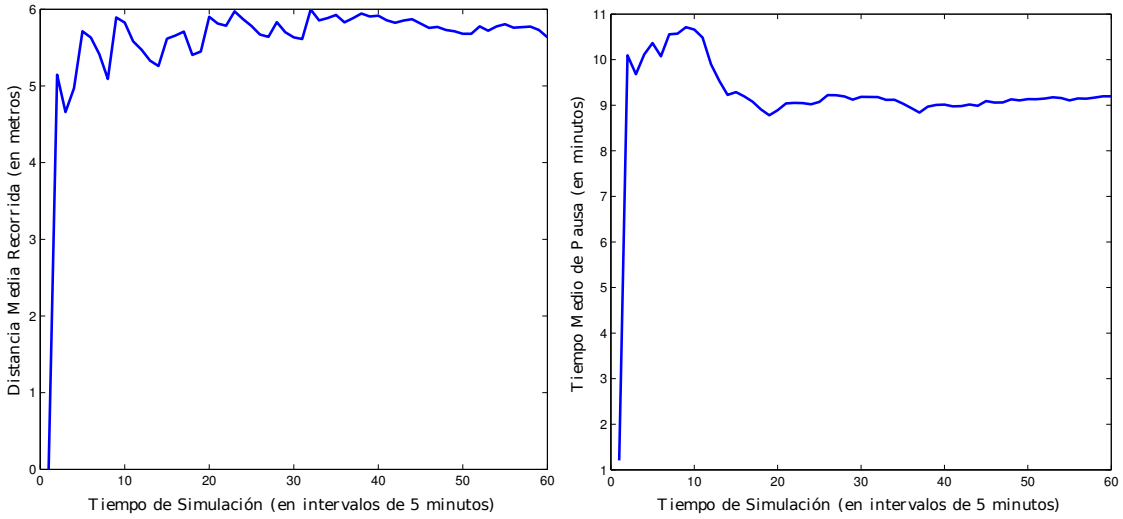


Figura 5.6: Distancias medias (izquierda) y tiempos de pausa medios (derecha) de los nodos.

siguientes distancias:

- Anillo de grado uno: $r_1 = 19\ metros$.
- Anillo de grado dos: $r_2 = 30\ metros$.

De esta forma, en la figura 5.7 se muestran el resultado obtenido tras la aplicación del mecanismo de búsqueda de caminos durante 4 intervalos de tiempo consecutivos, es decir, durante 15 minutos. Se observa una topología cambiante que representan este tipo de redes.

Una vez establecidos los enlaces se ejecutará el algoritmo de encaminamiento implementado (en nuestro caso se ha seleccionado el algoritmo de K-Shortest Path con 3 posibles caminos por demanda), el cual, a partir de los pesos asignados a cada enlace, generará las rutas entre los nodos con demanda. El peso de los enlaces se habrá asignado en función del grado (anillo de cobertura) al que pertenezcan. El número de posibles caminos a optimizar aumenta exponencialmente la complejidad del problema.

Resultados de optimización finales:

El último paso del simulador consiste en optimizar los flujos de demanda en función de unos costes. Para el ejemplo se ha utilizado una función de costes que penaliza el

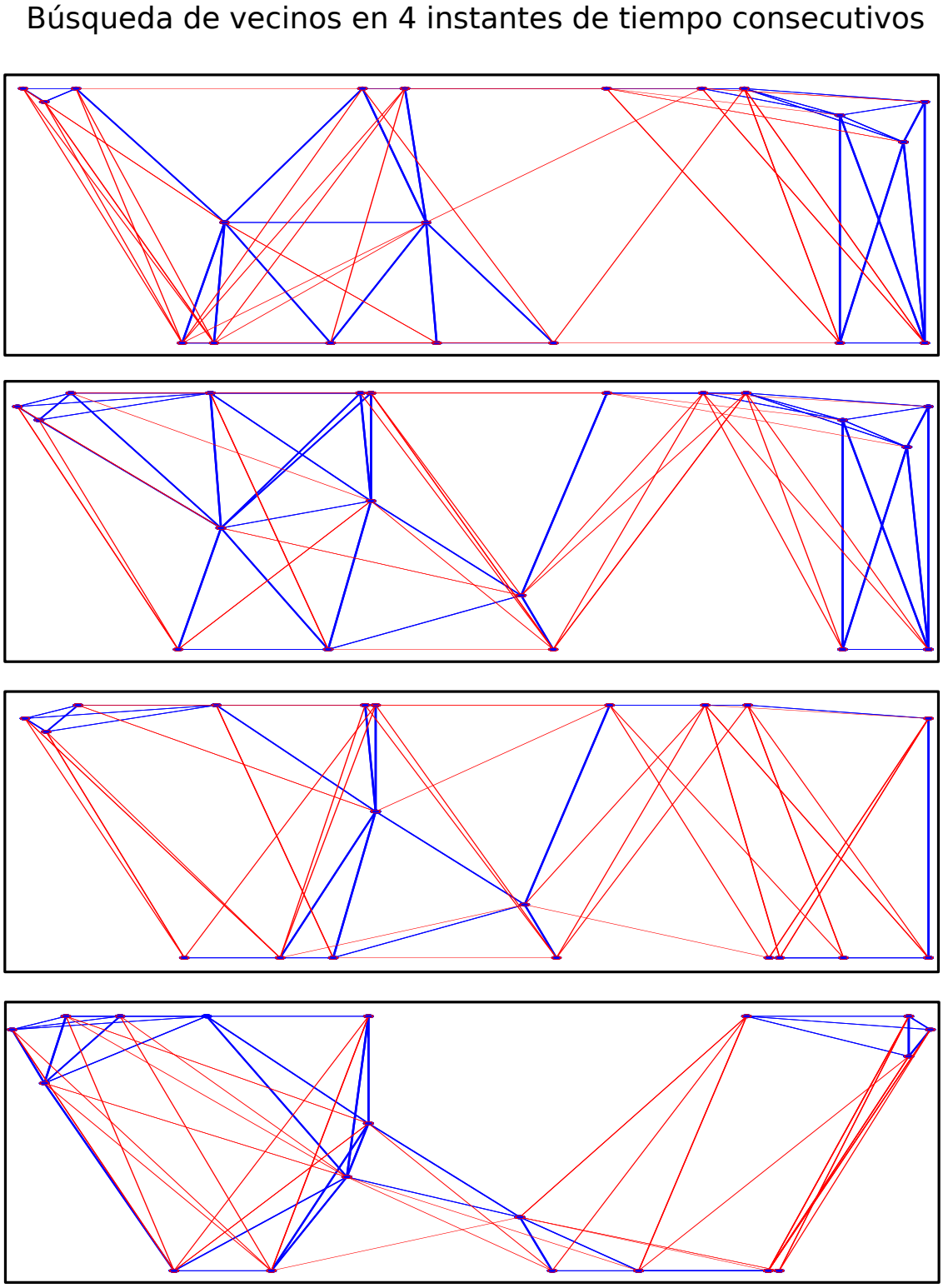


Figura 5.7: Evolución de la topología de red (enlaces establecidos) en un intervalo de 15 minutos.

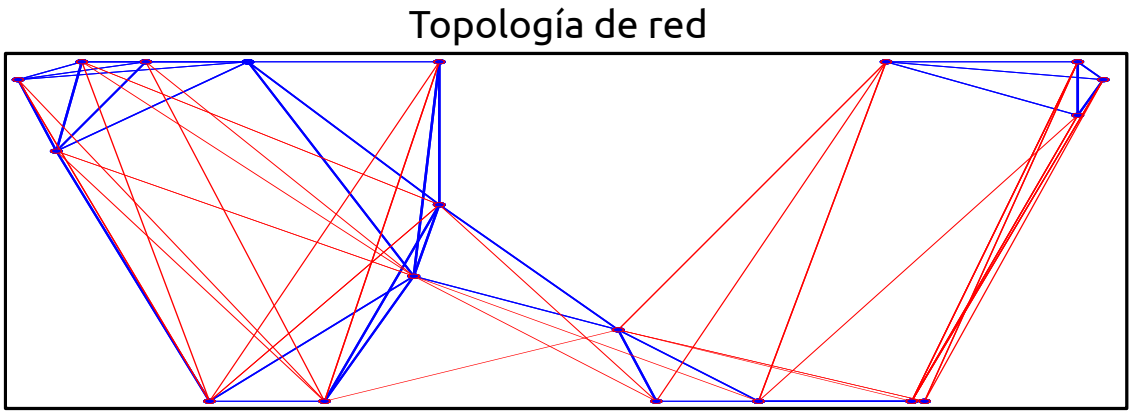


Figura 5.8: Enlaces establecidos en un intervalo de tiempo.

número de saltos. Se ha optado por esta opción tanto por sencillez como por la existencia real de mecanismos que buscan la minimización de este factor.

La topología sobre la que se realiza la optimización se muestra en la figura 5.8 que se corresponde con el último de los instantes de tiempo de la figura 5.7. El operador podría realizar esta optimización en todos los instantes de la simulación que se deseen, a costa de un sacrificio en el tiempo y los recursos de computación.

El simulador asigna las demandas de forma aleatoria, únicamente se indica el tamaño y número de estas que se desea que haya en la red. Esto supone una limitación y plantea problemas con la aparición de nodos aislados, ya que si en alguna de las demandas interviene uno de estos nodos, la función de optimización, lógicamente, no podrá encontrar una solución válida. Al tratarse redes con topologías cambiantes dificulta la simulación de este tipo de situaciones. Dada la complejidad del problema, y el coste computacional que implican el uso de herramientas de optimización como `fmincon`, la situación se agrava con la existencia de múltiples demandas simultáneas entre todos los nodos de la red.

Por tanto se ha realizado una optimización sencilla del ejemplo que sirva de cierre de éste e ilustre esta última parte del simulador.

En la prueba realizada se ha dimensionado la capacidad de los enlaces de la siguiente manera:

- Enlaces de grado uno (azules): 54Mbps ($r < r_1$).
- Enlaces de grado dos (rojos): 6Mbps ($r_1 < r < r_2$).

Introduciendo unas demandas entre nodos seleccionados aleatoriamente con las siguientes características:

- Una demanda $h_1 = 9Mb$.
- Una demanda $h_2 = 11Mb$.

Se introducen dos demandas entre nodos, a priori no se tiene conocimiento de entre que nodos se producen esas demandas. Sin embargo, a la vista del resultado (ver figura 5.9), se ha seleccionado los nodos N1 y N2 como origen y destino de la demanda h_1 y los nodos N3 y N4 para la demanda h_2.

Al utilizarse una función de costes que no penaliza la sobrecarga de enlaces, el simulador ocupa los enlaces al máximo, de forma que vemos enlaces en rojo, que representan a aquellos enlaces con una carga superior al 75% de su capacidad.

Para entender mejor el resultado obtenido se etiquetan las partes mas importantes del proceso en la figura 5.9. Entre los nodos de color azul (N1 y N2), correspondientes a la demanda de 9Mb, se observa la ocupación completa del enlace 1 y encaminamiento del tráfico restante por los enlaces 2 y 3. El color del enlace 3 es amarillo puesto que se encuentra con una carga media, comprendida entre el 25 y 75 por ciento, el enlace 2 asumirá el mismo flujo que el 3, sin embargo aparece en verde (carga inferior al 25%) ya que se trata de un enlace de grado uno, y por lo tanto tiene mayor capacidad (54Mbps).

Los nodos de color naranja (N3 y N4) corresponden a la segunda demanda (11Mb). En este caso el enlace 4 y 6 están por encima del 75% de carga, ya que se trata de una demanda superior a la anterior, y además ambos flujos atraviesan enlaces de grado dos con menor capacidad (6Mbps) y harán que los enlaces estén cargados. El enlace 5, al igual que el 2, es un enlace de grado uno.

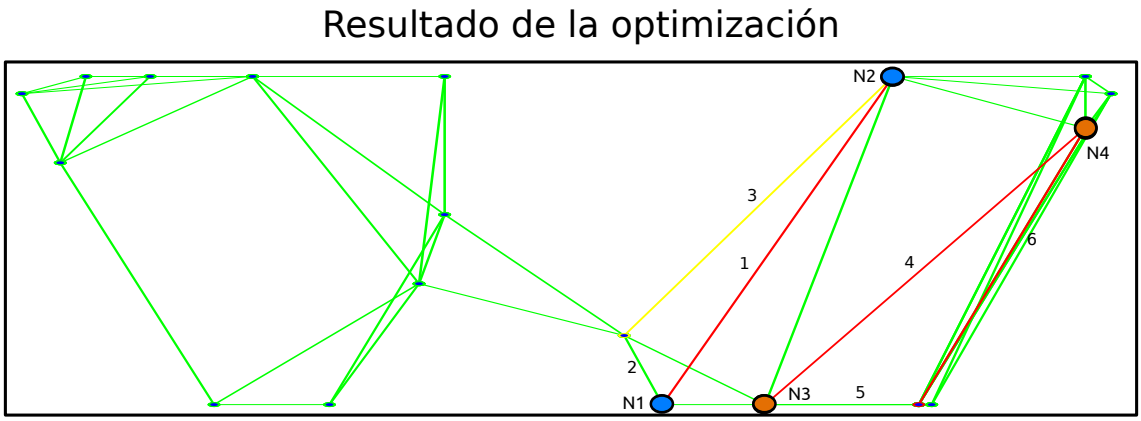


Figura 5.9: Enlaces establecidos en un intervalo de tiempo.

Capítulo 6

Conclusiones y Futuras Líneas de Trabajo

En el presente capítulo se desarrollan, en un primer apartado, las conclusiones derivadas del desarrollo de este proyecto. En el siguiente apartado, se realizará una visión de futuro respecto a las posibles líneas futuras por la que continuar trabajando.

6.1. Conclusiones

Las conclusiones obtenidas durante la realización del trabajo para este proyecto son las siguientes:

- En este proyecto se han sentado las bases para la creación de un simulador completo de redes ad hoc centrado en la optimización de tráfico en función de restricciones de coste. Puede resultar de gran utilidad contar con una herramienta como la que se plantea para realizar estudios y análisis de este tipo de redes, debido a la complejidad de su estudio por la gran cantidad de parámetros de un sistema completo.

- Se trata de un proyecto de gran envergadura con tres partes diferenciadas a nivel teórico, que han de integrarse en un sistema final, para obtenerse la funcionalidad completa. Al tratarse de un proyecto tan complejo se ha intentado hacer todo de forma modular para facilitar su escalabilidad y funcionalidad en todos los niveles.

  Esta modularidad responde a la gran cantidad de mejoras y ampliaciones (modelos

de movimiento, algoritmos de encaminamiento, funciones de coste, tipos de escenario, etc.) aplicables a cada una de las partes que se han mencionado durante este documento.

- Se ha visto que los modelos de movilidad son un campo importante a la hora de prever el comportamiento de la red. Con este trabajo se ofrece una herramienta que permite al diseñador probar en distintas situaciones los modelos diseñados.

- De igual forma ocurre con la parte de encaminamiento, ya que se encuentra en constante evolución debido a los requisitos que demandan este tipo de redes.

- Finalmente la optimización de este tipo de escenarios representa una gran complejidad. Es un campo muy importante para ajustar los parámetros de dimensionamiento de la red, posibilitar la toma de decisiones por parte de los nodos, etc. Sin embargo, el coste computacional de los mecanismos de optimización y la capacidad de procesamiento de los dispositivos actuales, hacen la investigación de este campo, la más interesante de las tres.

- Como conclusión a este proyecto podemos ofrecer una experiencia real en la que el coste computacional de la utilización de nuestra herramienta completa no permitiría su implementación en un nodo estándar de los que se encuentran en el mercado. En todo caso se podría disponer de nodos especiales, con mayores capacidades que pudiesen realizar la optimización de forma centralizada. O en su defecto, se podría introducir en la herramienta heurísticos para la optimización que tuviesen un menor consumo de recursos en computación y acelerasen el proceso.

6.2. Futuras Líneas de Trabajo

Finalmente se presentan algunas de las posibles líneas futuras de trabajo a seguir de cara a la continuación del trabajo realizado en este proyecto:

- Relativo a los modelos de movimiento:

    - Extender el número de modelos implementados permitiría aumentar la funcionalidad del simulador.

  - Validar todos los modelos incluidos en el simulador es vital para obtener buenos resultados. Esto se conseguiría mediante la posibilidad de incluir trazas reales o trazas obtenidas de comparativas cuantitativas con modelos ya validados.
  - Desarrollar una herramienta gráfica el diseño y parametrización de los modelos de movimiento y que proporcione los parámetros de inicialización necesarios para nuestro sistema.

- Concerniente al encaminamiento:

  - Implementar otros mecanismos reales de establecimiento de enlaces: Análisis y estudio de prestaciones de cobertura, capacidad de número de conexiones, etc.
  - Implementar en el simulador otros protocolos de encaminamiento más propios de redes ad hoc y que permitieran la introducción de análisis de protocolos multicamino reales.

- Concerniente a la optimización:

  - Estudio más detallado de los algoritmos de optimización que incorpora la función `fmincon` para averiguar la influencia de todos ellos en el resultado final, ya que la inicialización y elección de heurísticos diferentes pueden dar soluciones finales con ligeras diferencias con restricciones finales.
  - Asimismo, la implementación y estudio de nuevas funciones de coste que permitan modelar otros requisitos.

- Respecto a la herramienta final:

  - Trabajar en una interfaz gráfica que permita introducir todo los parámetros de forma sencilla y ordenada, seleccionar modelos y algoritmos que se deseen utilizar y presentar los resultado de forma gráfica en un entorno amigable para usuarios no familiarizado con estos sistemas.
  - Realizar un validación mediante simulaciones de escenarios reales de los que se conozca su comportamiento, mediante datos obtenidos de mediciones reales.

Bibliografía

[1] U. U. UC3M, UAH, "Proyecto t2c2: tecnologías telemáticas para colaboración ciudadana." http://adscom.it.uc3m.es/~t2c2/index.html, 2009. [En línea; acceso el 5-julio-2011].

[2] F. Spitzer, *Principles of random walk*, vol. 34. Springer Verlag, 2001. ISBN:9780387951546.

[3] J. Yoon, M. Liu, and B. Noble, "Random waypoint considered harmful," vol. 2, pp. 1312–1321, 2003.

[4] C. Bettstetter, "Smooth is better than sharp: a random mobility model for simulation of wireless networks," 2001.

[5] P. Nain, D. Towsley, B. Liu, and Z. Liu, "Properties of random direction models," vol. 3, pp. 1897–1907, 2005.

[6] R. Bravo *et al.*, "Estudio de modelos de movimiento en interiores para aplicación en entornos wlan," 2007.

[7] M. aware Networking, "Tlw (truncated levy walk) mobility model." http://research.csc.ncsu.edu/netsrv/?q=content/tlw-truncated-levy-walk-mobility-model. [En línea; acceso el 22-junio-2011].

[8] D. Shukla and S. Iyer, "Mobility models in ad-hoc networks," *KReSIT-IIT Bombay*, 2001.

[9] M. Frodigh, P. Johansson, and P. Larsson, "Formación de redes inalámbricas ad hoc—el arte de la formación de redes sin red," *Ericsson Review*, pp. 248–263, 2000.

[10] S. Basagni, M. Conti, S. Giordano, and I. Stojmenović, *Mobile ad hoc networking*. Wiley-IEEE press, 2004. ISBN:0471373133.

[11] C. Liu and J. Kaiser, "A survey of mobile ad hoc network routing protocols," 2005.

[12] V. Díaz and J. José, "Teoría del encaminamiento en redes ad hoc inalámbricas," 2007.

[13] A. Tanenbaum, *Computer networks*. Prentice Hall PTR, 2003. ISBN:9780130661029.

[14] J. Yen, "Finding the k shortest loopless paths in a network," *management Science*, vol. 17, no. 11, pp. 712–716, 1971.

[15] B. Karp and H. Kung, "Gpsr: greedy perimeter stateless routing for wireless networks," in *Proceedings of the 6th annual international conference on Mobile computing and networking*, pp. 243–254, ACM, 2000.

[16] U. d. V. Enriqueta Vercher, "Optimización convexa." `http://www.upv.es/deioac/Doctorado/eio/c17.html`, 2007-2008. [En línea; acceso el 3-julio-2011].

[17] P. Venkataraman, *Applied optimization with MATLAB programming*. New York : John Wiley and Sons, 2002. ISBN: 0471349585.

[18] M. Pióro and D. Medhi, *Routing, flow and capacity design in communication and computer networks*. Amsterdam [etc.] : Morgan Kaufmann, 2004. ISBN:0125571895.

[19] M. A. Branch and A. Grace, *MATLAB: Optimization Toolbox: user's guide*. Th Math Works Inc., 1996. ISBN:(no encontrado).

[20] I. The MathWorks, "Matlab." `http://www.mathworks.com/`, 2011. [En línea; acceso el 13-julio-2011].

[21] I. The MathWorks, "Optimization toolbox: fmincon." `http://www.mathworks.com/help/toolbox/optim/ug/fmincon.html`, 2011. [En línea; acceso el 16-junio-2011].

Apéndice A

Presupuestos

En este apéndice se realiza un análisis de este proyecto en términos económicos.

A.1. Planificación de tareas

A continuación se muestra una estimación de la duración de las distintas tareas llevadas a cabo durante el desarrollo de este proyecto. Estas tareas no se han llevado a cabo de forma cronológica.

En la tabla A.1 se presentan estas tareas, junto con el número de horas estimadas para su realización.

Tarea	Número de horas
Investigación, documentación y análisis del problema	120 horas
Despliegue y configuración del entorno de trabajo	2 horas
Estudio, diseño y desarrollo del bloque de movilidad de nodos	200 horas
Estudio, diseño y desarrollo del bloque de encaminamiento	100 horas
Estudio, diseño y desarrollo del bloque de optimización	150 horas
Estudio y diseño del sistema completo	70 horas
Integración del sistema completo	50 horas
Realización de pruebas	100 horas
Obtención de resultados	100 horas
Redacción y corrección de la memoria	140 horas

Tabla A.1: Tareas del proyecto

A.2. Costes

En este apartado se determinan los costes debidos a los siguientes factores: personal, material (dividiéndose en equipo y licencias) y costes indirectos

A.2.1. Personal

Para la realización del proyecto, se ha necesitado cubrir las tareas que realizarían los especialistas que se detallan en la tabla A.2, junto con el precio de la mano de obra por hora

Cargo	Coste (por hora)
Jefe de proyecto	45,00€
Analista	35,00€
Diseñador	40,00€
Gestión de calidad y pruebas	35,00€
Programador	25,00€

Tabla A.2: Salarios de especialistas

Aplicándose estos costes al número de horas dedicadas por cada especialista en función de las actividades y las tareas encomendadas, se puede calcular el coste total del personal requerido, a través de la siguiente la tabla A.3.

El coste del personal asciende a 33.650,00€.

A.2.2. Material

En este apartado se detalla el coste de los materiales empleados durante la realización del proyecto. Estos costes se pueden dividir en equipo y licencias.

Equipo

El equipo utilizado consta de :

Actividad	Jefe de proyecto	Analista	Diseñador	Gestor de pruebas	Programador	Horas totales
Investigación, documentación y análisis del problema	30	90				120
Despliegue y configuración del entorno de trabajo					2	2
Estudio, diseño y desarrollo del bloque de movilidad de nodos	1	9	70		120	200
Estudio, diseño y desarrollo del bloque de encaminamiento	1	9	20		70	100
Estudio, diseño y desarrollo del bloque de optimización	1	9	60		80	150
Estudio y diseño del sistema completo	5	25	40			70
Integración del sistema completo			10		40	50
Realización de pruebas				100		100
Obtención de resultados		60			40	100
Redacción y corrección de la memoria	5	65	50		20	140
Horas totales	42	256	250	100	372	10320
Costes	**1.890,00€**	**8.960,00€**	**10.000,00€**	**3.500,00€**	**9.300,00€**	**33.650,00€**

Tabla A.3: Coste del personal

- Ordenador personal portátil. Valorado en 600€.

Licencias

En la tabla A.4 analizamos el coste de las licencias que hemos necesitado en la elaboración del proyecto.

Concepto	Licencias	Coste/licencia	C.A.	Coste
Ubuntu Linux	1	0,00€		0,00€
MATLAB (Licencia comercial individual)	1	500,00€	1/6	75,00 €
MATLAB (Toolbox de optimización)	1	200,00€	1/6	30,00 €
Latex	1	0,00€		0,00€
Total				**105,00€**

Tabla A.4: Coste de licencias

Es coste total en licencias es de 105,00€.

A.2.3. Transporte

Durante la realización del proyecto se utilizó como medio de transporte un vehículo privado con un coste de 300€.

A.2.4. Costes indirectos

En la tabla A.5 se resumen el resto de costes, derivados del uso de instalaciones durante los dos meses y medio que duro el proyecto.

El montante asciende a 2.280€ en este apartado.

[1]La cobertura por bajas se aplica en caso de tener que contratar personal para cubrir bajas por enfermedad o incapacidad.

[2]El seguro a todo riesgo incluye cobertura de personal y bienes materiales.

Concepto	Coste
Alquiler del local	720,00€
Luz y agua	180,00€
Servicio de limpieza	180,00€
Teléfono fijo y conexión a internet	210,00€
Cobertura por bajas[1]	870,00€
Seguro a todo riesgo[2]	120,00€
Total	**2.280,00€**

Tabla A.5: Costes indirectos

A.3. Resumen

Teniendo en cuenta todos los costes relatados anteriormente, el coste total necesario para realizar el proyecto asciende a 36.935,00€, tal y como se muestra en la tabla A.6.

Concepto	Cantidad
Mano de obra	33.650,00€
Equipo	600,00€
Licencias	105,00€
Transporte	300,00€
Costes indirectos	2.280,00€
Total	**36.935,00€**

Tabla A.6: Costes indirectos

Esta cantidad no tiene en cuenta ni el análisis del riesgo ni los impuestos.

A.3.1. Totales

Finalmente, en la tabla A.7 se desglosa el balance final del coste del proyecto.

Concepto	Cantidad
Coste total	36.935,00€
Riesgo (20 %)	7387,00€
Beneficio (20 %)	7387,00€
Total sin I.V.A.	51.709,00€
I.V.A. (18 %)	9.307,62€
Total	**61.016,62€**

Tabla A.7: Presupuesto total

Teniendo en cuenta los costes desglosados en los apartados anteriores, el presupuesto total de este proyecto asciende a la cantidad de **SESENTA Y UN MIL DIECISÉIS CON SESENTA Y DOS** euros.

Leganés, a 21 de Julio de 2011

El ingeniero proyectista

Fdo. Pablo de la Fuente Nuez

Apéndice B

Estructura de los Resultados Intermedios

En este apéndice se realiza una presentación de las estructuras de datos que utiliza el simulador durante el proceso.

B.1. Introducción

Este apéndice da una idea del tipo de datos que está manejando el programa. Da una visión del funcionamiento del simulador a un nivel más bajo, para ello se hace uso del ejemplo utilizado en el capítulo 4.

B.2. Estructuras de datos del Bloque de Movilidad

El bloque de movilidad hace uso de vectores y matrices para almacenar las características particulares de cada nodo.

Esta es la estructura (ver figura B.1) utilizada para el modelo Random Waypoint modificado para un intervalo de tiempo, agrupa toda la información necesaria para realizar la actualización de los nodos en el siguiente intervalo de tiempo.

Las dos primeras columnas indican la posición actual del nodo (coordenadas X e Y) en el escenario, la tercera y cuarta columna muestran la dirección que llevan los nodos en movimiento, por otro lado la quinta columna indica el tiempo de pausa de los nodos

```
>> [POS DIR P D V]

ans =

   10.0000    5.0000   -0.7714   -0.6364         0    6.0000    0.9755
    7.0000    1.0000   -0.6267    0.7792         0    4.0000    0.9790
    7.0000    3.0000         0         0    3.6066         0         0
    2.0000   10.0000    0.5344   -0.8452         0    9.0000    0.8681
    4.0000    8.0000   -0.8947   -0.4468         0    1.0000    0.9275
```

Figura B.1: Estructura de datos referente a los modelos de movilidad.

```
adyacencia =

     0     1     1     9     9
     1     0     1     0     9
     1     1     0     9     0
     9     0     9     0     1
     9     9     0     1     0
```

Figura B.2: Estructura de datos referente a la búsqueda de vecinos.

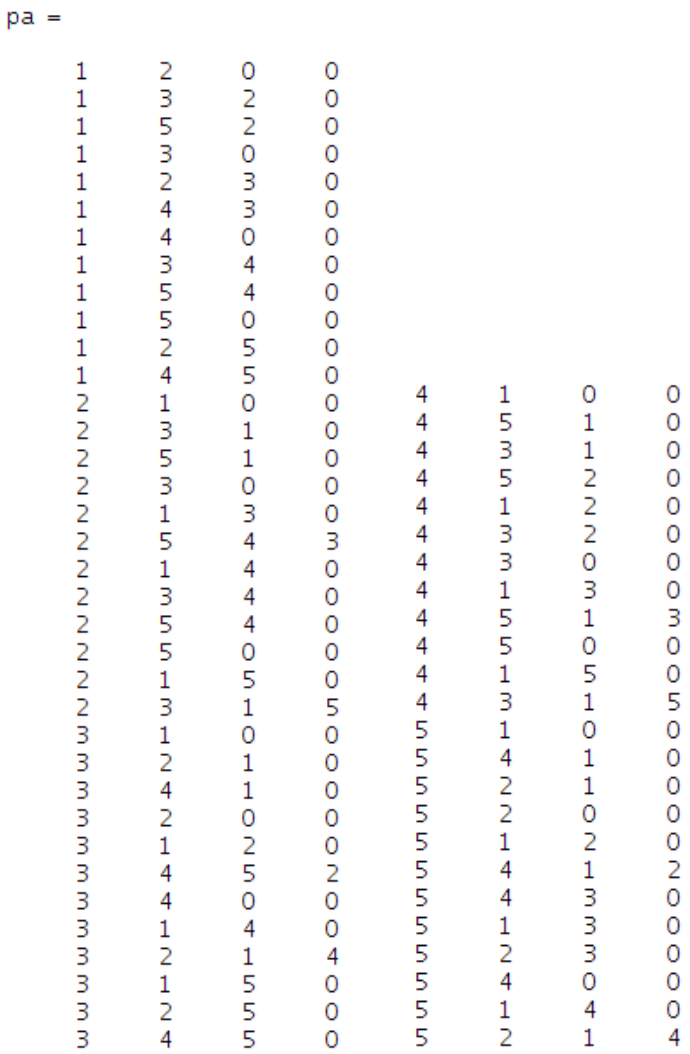
que están parados (el resto de nodos van a cero en esta posición) y por último, las dos columnas restantes completan la información de los nodos en movimiento, estas son, respectivamente, las distancias y velocidades.

B.3. Estructuras de datos del Bloque de Encaminamiento

En este bloque se realiza la búsqueda de vecinos y caminos entre los nodos. Por lo tanto se genera la matriz de adyacencia (con los costes asociados) y las rutas entre nodos con demandas.

En el ejemplo del capítulo 4 se simulaba una red de cinco nodos para facilitar la comprensión del mismo, se eligió como método de búsqueda de vecinos el de *anillos de cobertura*, el cual genera una matriz de adyacencia como la de la figura B.2. Se trata de una matriz de cinco filas por cinco columnas. Los valores de esta representan el coste del enlace (en función del grado del anillo al que pertenezcan).

Una matriz de adyacencia generada con el mecanismo de selección *K nodods* da



(a) Primera parte. (b) Continuación de la matriz.

Figura B.3: Estructura de datos referente a la búsqueda de caminos.

como resultado una matriz de las mismas dimensiones pero reflejando en las posiciones que representen un enlace, la distancia entre el par de nodos que lo forman.

La otra estructura de datos generada (ver figura B.3) indica los caminos que existen entre nodos con demanda, para el ejemplo se usó una demanda base entre todos los nodos (con valor 0.4 unidades) con cuatro demandas pico (con valores 2, 5, 6 y 7 unidades). A parte, se decidió que el algoritmo *K-Shortest Path* buscase un máximo de tres caminos por demanda, la estructura de datos resultante se puede ver en la figura B.3.

```
enlaces =

     1     2
     1     3
     1     4
     1     5
     2     3
     2     5
     3     4
     4     5
```

Figura B.4: Estructura de datos referente a la búsqueda de vecinos.

B.4. Estructuras de datos del Bloque de Optimización

Las estructuras de datos concernientes a este bloque son las que suponen un mayor esfuerzo en diseño, almacenamiento y cómputo. Junto con las capacidades, demandas y caminos establecidos por el bloque de encaminamiento (capítulo 3) se utiliza una función de costes para minimizar.

El primer paso que se realiza es el parseo de nodos extremos a etiquetado de enlaces (ver figuras B.4). Para este bloque se necesita realizar un etiquetado de los enlaces, puesto que la optimización se centra sobre los flujos de tráfico que pasan por éstos, de manera que utilizando la posición (en fila) que ocupan cada par de enlaces en B.4 sabremos a que enlace nos referimos, se desarrolla un tabla con los enlaces existentes en la red a partir de la matriz de adyacencia (ver figura B.2).

Haciendo uso de los caminos entre nodos (ver figura B.3), que son los flujos de demanda entre nodos y del etiquetado de enlaces (ver figura B.4) se construye la matriz `A` buscando los flujos que pasan por un enlace dado. Se compara el par de nodos de B.4 con todos los saltos de todos los caminos de B.3 resultando una matriz como la de la figura B.5.

El vector `b` corresponde a las capacidades de los enlaces, las cuales no se pueden sobrepasar. Esta estructura se crea a partir del listado de enlaces (ver figura B.4) y de la matriz de adyacencia (ver figura B.2), para tener conocimiento de que tipo de enlace se trata (en el ejemplo, grado uno o dos) y asignar una capacidad en consecuencia.

Por otro lado, se genera la otra gran estructura de datos necesaria para utilizar como

```
A =

  Columns 1 through 11

     1     0     0     0     1     0     0     0     0     0     1
     0     1     0     1     0     0     0     1     0     0     0
     0     0     0     0     0     1     1     0     0     0     0
     0     0     1     0     0     0     0     0     1     1     0
     0     1     0     0     1     0     0     0     0     0     0
     0     0     1     0     0     0     0     0     0     0     1
     0     0     0     0     0     1     0     1     0     0     0
     0     0     0     0     0     0     0     0     1     0     0

  Columns 12 through 22

     0     1     0     0     0     1     0     1     0     0     0
     0     0     1     0     0     1     0     0     0     0     0
     1     0     0     0     0     0     0     1     0     0     0
     0     0     0     1     0     0     0     0     0     0     0
     0     0     1     0     1     0     0     0     1     0     0
     0     0     0     1     0     0     1     0     0     1     1
     0     0     0     0     0     0     1     0     1     0     0
     1     0     0     0     0     0     1     0     0     1     0

  Columns 23 through 33

     1     0     0     1     0     0     1     0     0     0     1
     0     1     1     0     0     0     1     0     0     1     0
     0     0     0     0     1     0     0     0     0     1     1
     1     1     0     0     0     0     0     0     0     0     0
     0     1     0     1     0     1     0     0     0     0     1
     0     0     0     0     0     0     0     1     0     0     0
     0     0     0     0     1     0     0     1     1     0     0
     0     0     0     0     0     0     0     1     0     0     0

  Columns 34 through 44

     0     0     0     0     0     0     0     1     0     0     0
     1     0     0     0     0     1     0     0     0     0     1
     0     0     0     1     0     0     0     1     0     0     1
     1     0     0     0     1     0     0     0     0     0     0
     0     1     0     0     0     0     0     0     1     0     0
     0     1     0     0     0     0     1     0     0     0     0
     0     0     1     0     0     1     0     0     1     1     0
     0     0     1     0     1     0     1     0     0     0     0

  Columns 45 through 55

     0     0     0     0     0     0     1     0     1     1     0
     1     0     0     1     0     0     0     0     0     0     0
     0     0     1     0     0     1     0     0     0     1     0
     1     0     1     1     1     0     0     0     1     0     0
     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     1     1     0     0     0
     0     0     0     1     0     0     0     0     0     0     1
     1     1     0     0     0     1     0     0     0     1     1

  Columns 56 through 60

     0     0     0     0     1
     1     0     0     0     0
     0     0     0     1     1
     1     0     0     1     0
     0     1     0     0     0
     0     1     0     0     1
     0     0     0     0     0
     0     0     1     0     0
```

Figura B.5: Estructura de datos referente a las variables de la función de restricción de desigualdad.

```
b =

    54    54     6     6    54     6     6    54
```

Figura B.6: Estructura de datos referente a las variables de las estructuras de desigualdad.

entrada de la función `fmincon` que es `Aeq` (ver figuras B.7 y B.8). Indica la demanda entre nodos y los flujos pertenecientes a esa demanda. De modo que con el listado de todos los flujos (ver figura B.3) y el número de caminos por demanda se crea esta matriz.

Acompañando a `Aeq` está el vector `beq` (ver figura B.9) que indica el valor de las demandas entre nodos.

Con estos datos `A`, `b`, `Aeq` y `beq`, junto con la función de costes, el vector de valores iniciales de `x` y sus límites, se ejecuta la función `fmincon`. En caso de éxito arrojará la información que muestra la figura B.10 y la solución que cumple el problema planteado (ver figura B.11).

Se realiza un tratamiento de los resultados para presentar la información de forma útil, para ello se utiliza una estructura como la de la figura B.12, que indica la cantidad de demanda que pasa cada enlaces. La solución de la `fmincon` propone una solución para la cantidad de demanda entre dos nodos que debe asumir cada flujo perteneciente a dicha demanda, de forma que, sabiendo los flujos que pasan por cada enlace, se puede calcular el total absorbido por un enlace. Por último, se calcula la carga (en porcentaje) que esta soportando cada nodo (ver figura B.13) a partir del resultado anterior y la capacidad de cada enlace.

```
Aeq =

  Columns 1 through 15

     1     1     1     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     1     1     1     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     1     1     1     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     1     1     1     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     1     1     1
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0

  Columns 16 through 30

     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     1     1     1     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     1     1     1     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     1     1     1     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     1     1     1     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     1     1     1
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0     0     0     0     0     0
```

Figura B.7: Estructura de datos referente a las variables de la función de restricción de igualdad.

Columns 31 through 45

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1	1	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Columns 46 through 60

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1	1	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Figura B.8: Estructura de datos referente a las variables de la función de restricción de igualdad (continuación).

```
beq =

  Columns 1 through 6

    0.4000    6.0000    0.4000    0.4000    0.4000    0.4000

  Columns 7 through 12

    0.4000    2.0000    7.0000    0.4000    0.4000    0.4000

  Columns 13 through 18

    5.0000    0.4000    0.4000    0.4000    0.4000    0.4000

  Columns 19 through 20

    0.4000    0.4000
```

Figura B.9: Estructura de datos referente a las variables de las estructuras de igualdad.

```
Local minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the default value of the function tolerance,
and constraints were satisfied to within the default value of the constraint tolerance.

<stopping criteria details>

Active inequalities (to within options.TolCon = 1e-06):
  lower      upper     ineqlin   ineqnonlin
    2
    3
    5
    6
    8
    9
   11
   12
   14
   15
   17
   18
   23
   24
   26
   27
   29
   30
   32
   33
   38
   39
   44
   45
   47
   48
   50
   51
   53
   54
   59
   60
```

Figura B.10: Salida de la ejecución de la función `fmincon`.

```
x =

  Columns 1 through 6

    0.4000   -0.0000   -0.0000    6.0000    0.0000   -0.0000

  Columns 7 through 12

    0.4000   -0.0000   -0.0000    0.4000   -0.0000    0.0000

  Columns 13 through 18

    0.4000    0.0000   -0.0000    0.4000    0.0000    0.0000

  Columns 19 through 24

    0.0010    0.1636    0.2354    2.0000   -0.0000   -0.0000

  Columns 25 through 30

    7.0000    0.0000         0    0.4000    0.0000   -0.0000

  Columns 31 through 36

    0.4000   -0.0000    0.0000    0.1538    0.1568    0.0894

  Columns 37 through 42

    5.0000    0.0000   -0.0000    0.2354    0.0010    0.1636

  Columns 43 through 48

    0.4000   -0.0000    0.0000    0.4000   -0.0000    0.0000

  Columns 49 through 54

    0.4000    0.0000   -0.0000    0.4000   -0.0000   -0.0000

  Columns 55 through 60

    0.0894    0.1538    0.1568    0.4000    0.0000   -0.0000
```

Figura B.11: Estructura de datos referente a la solución de la función `fmincon`.

```
matriz_flujos =

         0    0.0149    0.2464    0.9003    0.1846
         0         0    0.0267         0    0.5308
         0         0         0    0.2177         0
         0         0         0         0    0.0268
         0         0         0         0         0
```

Figura B.12: Estructura de datos referente a al tráfico optimizado.

```
matriz_sobrecarga =

     0     1     1     3     1
     0     0     1     0     2
     0     0     0     1     0
     0     0     0     0     1
     0     0     0     0     0
```

Figura B.13: Estructura de datos referente a la carga de los enlaces.

Apéndice C

Implementación del Código

En este apéndice se realiza una presentación de las partes principales del código, funciones, parámetros de entra y salida.

C.1. Estructura General del Código

En la figura C.1 se muestra el esquema general con la estructura de ficheros del simulador de forma que se entienda cual es el cometido principal de cada uno de ellos. A continuación se detalla de manera general las entradas y salidas más importantes de cada una de las funciones:

- `[POS,DIR,V,D,P] = init_config(model, scenario)`

  Función para configurar el escenario y las condiciones de partida de los nodos. Tiene como entradas el modelo de movilidad a usar en la simulación y el tipo de escenario de prueba (necesario para la inicialización del modelo Truncated Levy Walk modificado). Devuelve el estado inicial de los nodos o las trazas del recorrido completo de los nodos para toda la simulación en caso de se haya seleccionado el modelo Truncated Levy Walk Modificado.

- `[Mobility] = TLW_MATLAB(alpha,beta,size_max,s_min,s_max,duration,b_c)`

  Función encargada de generar la traza de un nodo durante el tiempo de simulación, se encuentra detallada en el capítulo 2.

- `[POS_aux, distance] = update_position(POS, DIR, V, D, P, model, current_time, scenario)`

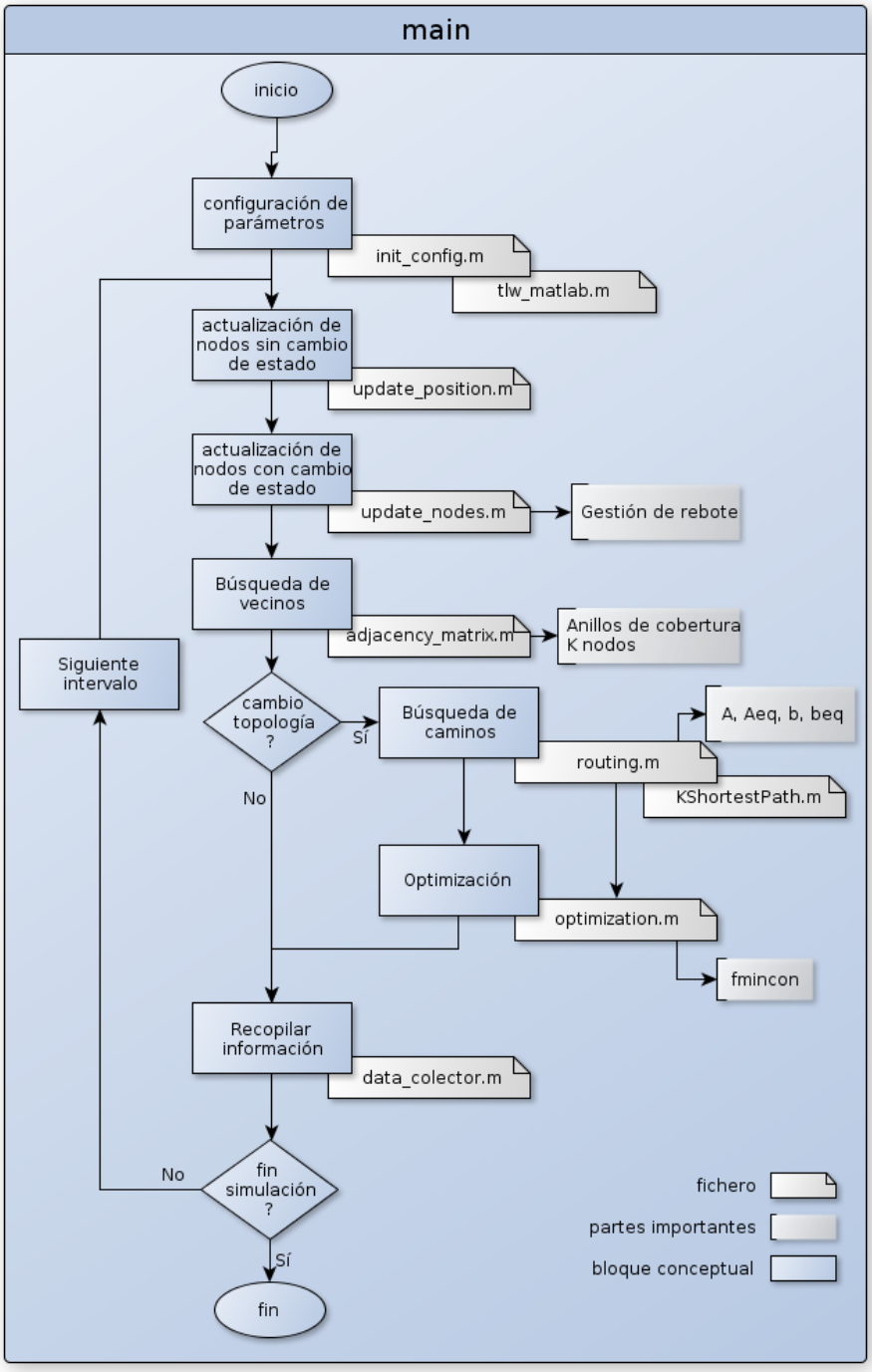


Figura C.1: Esquema de la estructura general del código.

Función que actualiza la posición de los nodos, tiene como entradas las características de los nodos y del escenario para moverlos, devolviendo las posiciones actualizadas y distancia recorridas.

- `[DIR, V, D, P] = update_nodes(POS, DIR, V, D, P, model, escenario, a_parado, a_parado_pared, a_movimiento, a_movimiento_pared)`

  Función que realiza la gestión de los nodos que cambian de estado. Es necesario conocer los nodos que cambian de estado y además se encuentran en una zona especial (borde o zona de tiendas) ya que requieren un tratamiento a parte. Recibe como entradas, la lista de los nodos que cambian de estado y aquellos que se encuentran en zona especial, junto con los parámetros que los caracterizan. Devuelve los parámetros actualizados de los nodos.

- `[adyacencia_aux] = adjacency_matrix(POS)`

  Función que realiza la búsqueda de vecinos y genera la matriz de adyacencia correspondiente. Posee como parámetro interno la elección de mecanismo de selección de vecinos. La entrada de la función son las posiciones de los nodos.

- `[A,b, Aeq, beq, pa] = routing(adj_matrix)`

  Función encargada de realizar la búsqueda de caminos y la adecuación de los datos de entrada para la optimización, por ello necesita como entrada la matriz de adyacencia, a parte de los parámetros internos necesarios, como son, las demandas (origen y destino y cantidad) y el número de caminos por demanda. Devuelve las matrices `A` y `Aeq` junto con sus vectores correspondientes `b` y `beq`, además de los caminos entre los nodos con demandas.

- `[shortestPaths, totalCosts] = kShortestPath(netCostMatrix, source, destination, k_paths)`

  Función que implementa el algoritmo (*K-Shortest Path*) encargado de la búsqueda de caminos, requiere como entradas, la matriz de costes (en nuestro caso será la matriz de adyacencia modificada), el nodo origen y destino, y el número de caminos máximo que se quiere buscar. Devuelve las rutas encontradas y el coste asignado.

- `[x,f,err] = optimization(A, b, Aeq, beq)`

  Función para hacer transparente la llamada a la `fmincon`, recibe los datos necesarios del problema a optimizar, a parte de la función de costes que se establece como parámetro propio de esta función. Devuelve las salidas que arroja la `fmincon`.

- `[] = data_colector()`

  Función para recoger la información que se va obteniendo durante la simulación y procesarla tal y como se ha ido explicando a lo largo de este documento.